EXHIBIT A

US009098526B1

(12) **United States Patent**
Tsao

(10) **Patent No.: US 9,098,526 B1**
(45) **Date of Patent: *Aug. 4, 2015**

(54) **SYSTEM AND METHOD FOR WIRELESS DEVICE ACCESS TO EXTERNAL STORAGE**

(71) Applicant: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(72) Inventor: **Sheng Tai Tsao**, Fremont, CA (US)

(73) Assignee: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **14/150,106**

(22) Filed: **Jan. 8, 2014**

Related U.S. Application Data

(63) Continuation of application No. 14/079,831, filed on Nov. 14, 2013, which is a continuation of application No. 10/726,897, filed on Dec. 4, 2003, now Pat. No. 8,606,880.

(51) **Int. Cl.**
G06F 15/16 (2006.01)
G06F 17/30 (2006.01)

(52) **U.S. Cl.**
CPC ***G06F 17/30194*** (2013.01)

(58) **Field of Classification Search**
USPC 709/219, 203, 226, 220, 200; 455/412, 455/899
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

8,606,880	B2 *	12/2013	Tsao	709/219
2001/0052073	A1 *	12/2001	Kern et al.	713/161
2002/0104080	A1 *	8/2002	Woodard et al.	717/176
2002/0133597	A1 *	9/2002	Jhingan et al.	709/228
2003/0072031	A1 *	4/2003	Kuwata et al.	358/1.15
2004/0033821	A1 *	2/2004	Slesak et al.	455/575.1
2004/0186750	A1 *	9/2004	Surbey et al.	705/4
2014/0089360	A1 *	3/2014	Tsao	707/827

* cited by examiner

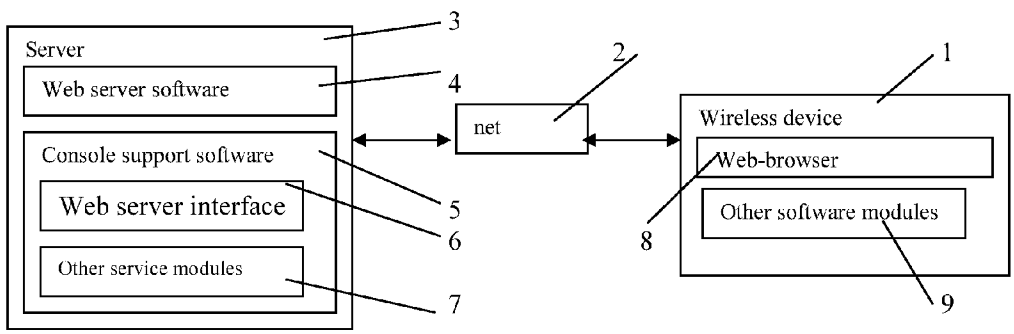
Primary Examiner — El Hadji Sall

(57) **ABSTRACT**

To meet the needs for storing larger volume personal information for user of wireless device, it is desire to provide extra storage space to the wireless device such as for cell phone or personal assistant device (PDA) etc due to the limited storage space that the wireless device has. Instant application disclosed a system and method for the wireless device to efficiently and effectively use remotely located storage space provided by a server for storing message or multimedia data such as videos, digital music, digital photo/picture.

20 Claims, 5 Drawing Sheets

Wireless devices supports in a simple environment



Wireless devices supports in a simple environment

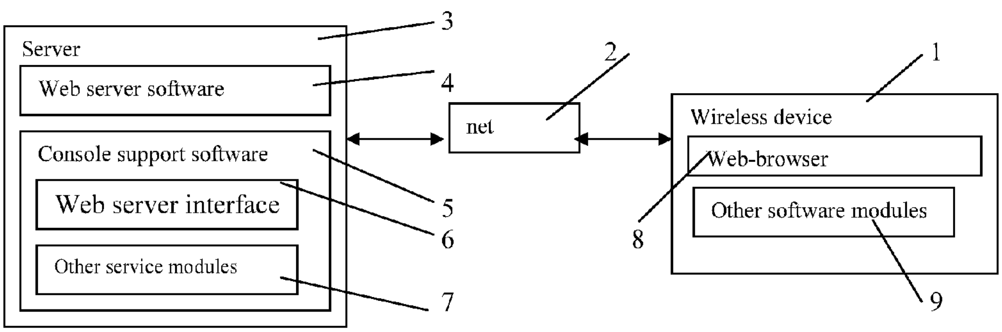


Fig. 1

Case 1:19-cv-01360-RGA Document 1-1 Filed 07/22/19 Page 4 of 12 PageID #: 14

Wireless devices access external storage through web browser

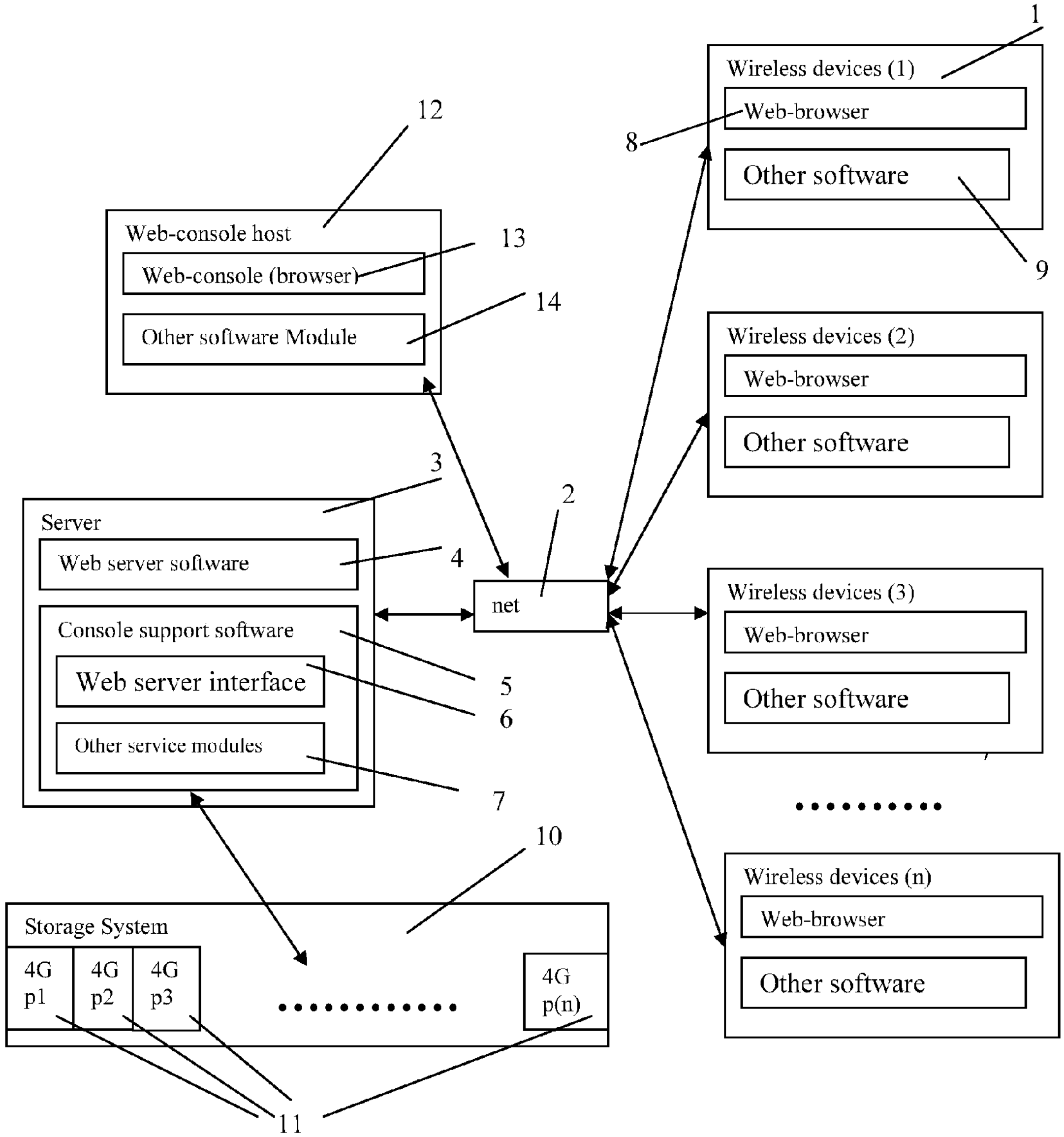


Fig. 2

Wireless out-band download

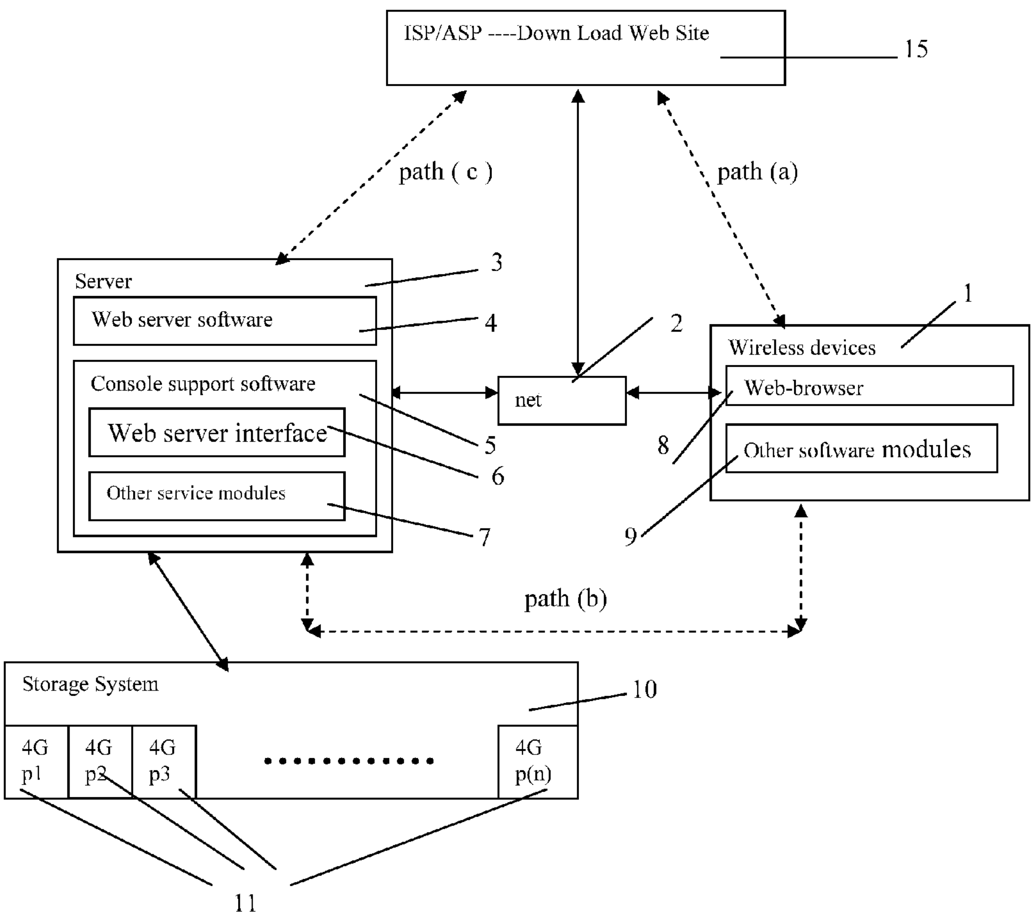


Fig. 3

Case 1:19-cv-01360-RGA Document 1-1 Filed 07/22/19 Page 6 of 12 PageID #: 16

The CCDSVM Support External Device for Huge Number of Wireless Device

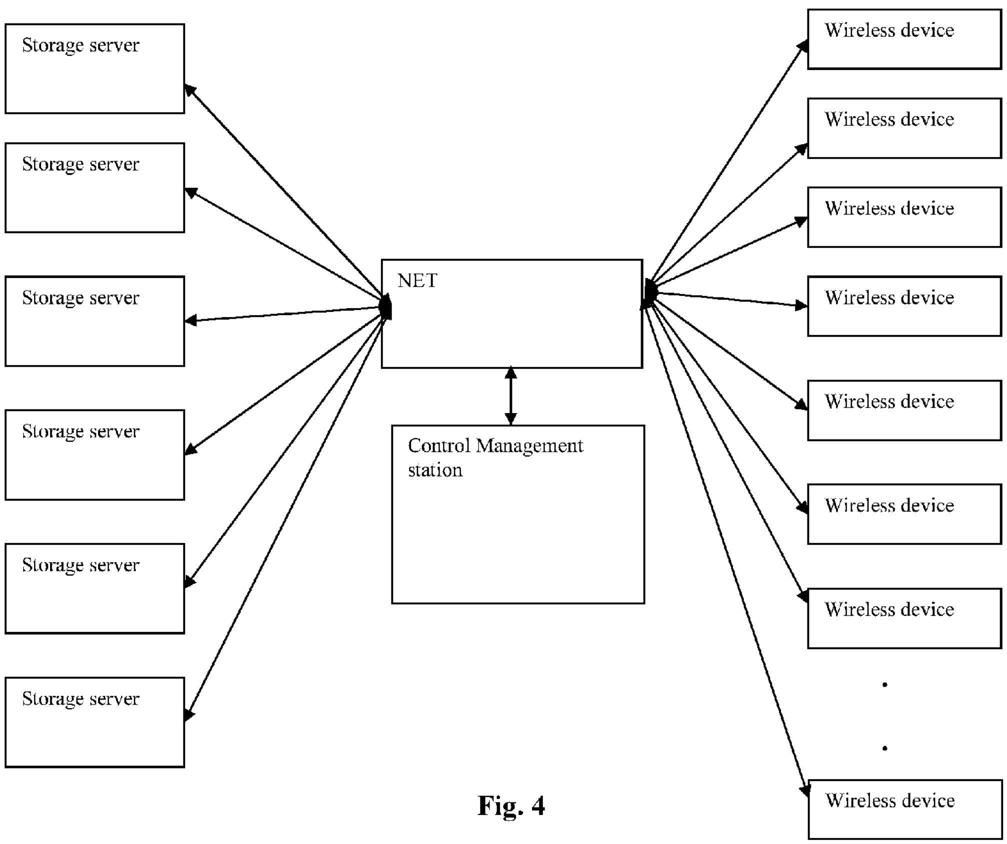


Fig. 4

Case 1:19-cv-01360-RGA Document 1-1 Filed 07/22/19 Page 7 of 12 PageID #: 17

A typical Computer system connected to network

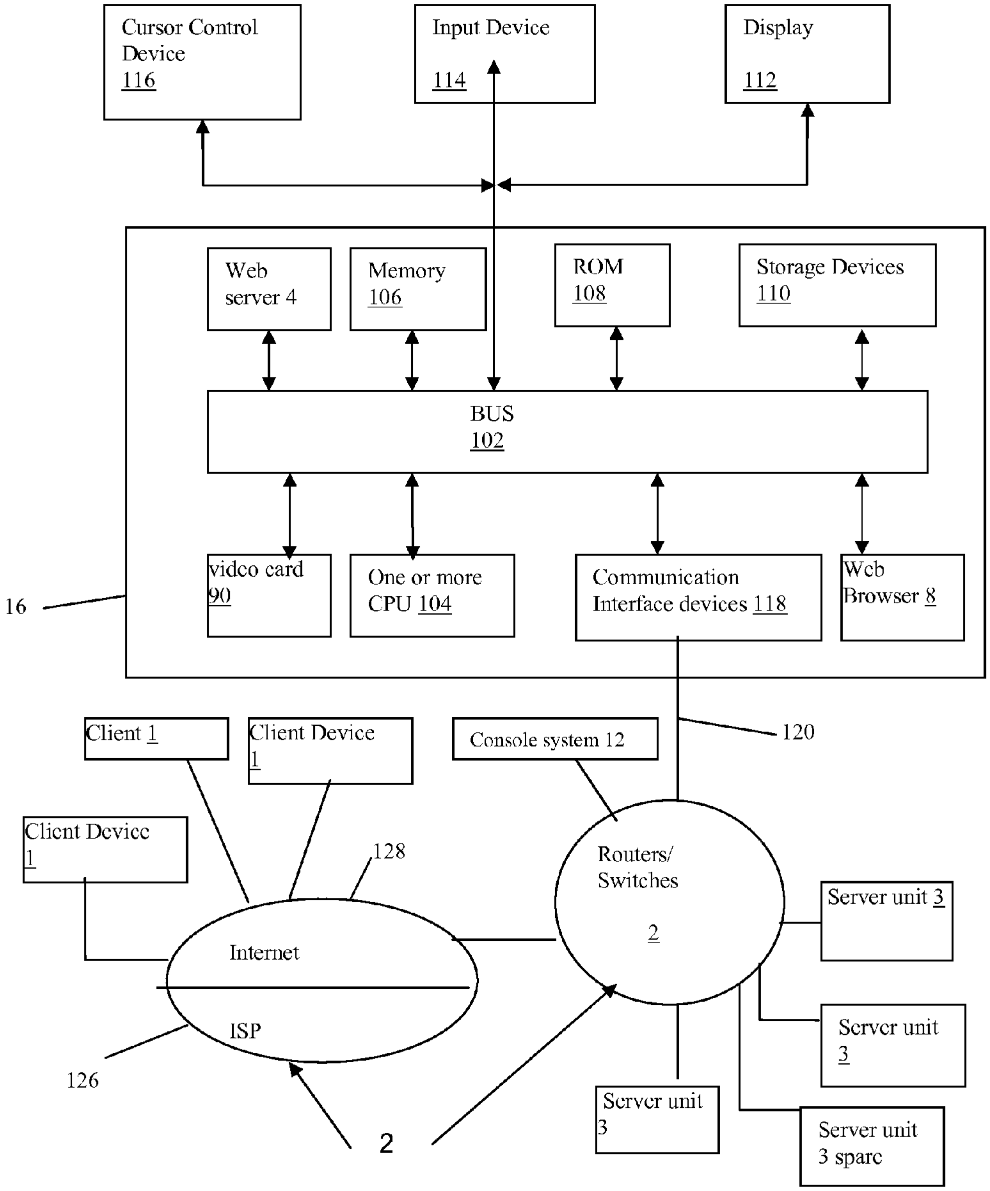


Fig. 5

Case 1:19-cv-01360-RGA Document 1-1 Filed 07/22/19 Page 8 of 12 PageID #: 18

SYSTEM AND METHOD FOR WIRELESS DEVICE ACCESS TO EXTERNAL STORAGE

CROSS REFERENCE TO PRIOR APPLICATION

This invention is a continuation application of the U.S. patent application Ser. No. 14/079,831 filed on Nov. 14, 2013 and now a U.S. Pat. No. 8,868,690, which in turn itself is a continuation application of U.S. patent application Ser. No. 10/726,897 filed on Dec. 4, 2003 and now a U.S. Pat. No. 8,606,880, which is referenced to a prior application No. 60/401,238 of "Concurrent Web Based Multi-task Support for Control Management System" filed on Aug. 6, 2002 and converted to application Ser. No. 10/713,904 which now is an U.S. Pat. No. 7,418,702. This application also referenced to an prior application No. 60/402,626 of "IP Based Distributed Virtual SAN" filed on Aug. 12, 2002 and converted to application Ser. No. 10/713,905 which now is an U.S. Pat. No. 7,379,990. All mentioned prior applications are herein incorporated by reference in their entirety for all purpose.

FIELD OF THE INVENTION

This invention focuses on how a wireless device can actually use external storage provided by a storage server.

BACKGROUND INFORMATION

Terminology:

The terminologies described in next few sections reflect the scope and are part of present invention.

The Internal Storage of a System:

The storage media such as hard disk drives, memory sticks, memory etc. is connected to a system directly through bus or a few inches of cable. Therefore, the storage media actually is a component of the system in an enclosure.

The External Storage of a System:

The external storage media is not a component of the system in a same enclosure. Therefore, they have to be connected through a connecting medium (e.g. a cable) such as Ethernet cable for IP based storage, Fiber channel cable for fiber channel storage, or such as wireless medium and etc. The storage media of an external storage could be magnetic hard disk drives, solid state disk, optical storage drives, memory card, etc. and could be in any form such as Raid which usually consists of a group of hard disk drives.

The Storage Partition, its Volumes, and the Corresponding File System:

To effectively use storage system, each storage device usually needs to be partitioned into small volumes. After the partition, each of the volumes can be used to establish a file system on it. To simplify the discussion herein, the term of the storage volume, its corresponding file system, and the term of the partition of the storage device are often used without differentiation.

CCDSVM:

It is an abbreviation for a central controlled distributed scalable virtual machine system. The CCDSVM allows a control management station to control a group of systems and provide distributed services to a client system on the Internet, the Intranet, and an LAN environment.

ISP & ASP:

The ISP refers to Internet service provider and the ASP refers to application service provider.

FIGURES

FIG. **1** illustrates an embodiment of the instant application, the FIG. **1** is the same as FIG. **1** of the prior application of the "Concurrent Web Based Multi-task Support for Control Management System" with an exception of replacing a console host with a wireless device.

FIG. **2** is the same as FIG. **1** of the above except that it shows a more detailed storage system controlled by a server. In addition, multiple wireless devices are presented for access to the storage system.

FIG. **3** shows a scheme of a wireless device downloading contents from an ISP/ASP or other web sites to an external storage allocated for the wireless device.

FIG. **4** is similar to the FIG. **1** of the prior application of the "IP Based Distributed Virtual SAN" with exception that each IP storage server provides file system as external storage for each of the wireless devices instead of providing IP based virtual SAN service. Also, each host in said FIG. **1** actually is replaced by a wireless device of FIG. **4**.

Unless specified, the programming languages and the protocols used by each software modules of instant application, and the computing systems used in this invention are assumed to be the same as described in the mentioned prior patent applications.

In addition, in the drawing, like elements are designated by like reference numbers. Further, when a list of identical elements is present, only one element will be given the reference number.

BRIEF DESCRIPTION OF THE INVENTION

Today users commonly face a problem of lack of storage capacity configured on their wireless devices such as cell phone or PDA, which are usually limited to 256 MB for the PDA and much less for the cell phone. To effectively solve this problem and let users own multiple gigabytes (GB) of storage for their wireless devices as well as allowing the users to use the GB storage for their multimedia applications, the storage of a server can be used as the external storage for the wireless devices. This technology has been briefly introduced in the prior patent applications.

Now let us examine how can the external storage actually be used by the wireless devices. First, let each server unit (e.g. the server **3** of the FIG. **2**) partitions its storage system into volume and each of the volumes will have multiple GB in size. Therefore, each user of the wireless devices can be exclusively assigned for access to a specific storage volume on a server unit. For example, if we need to provide each user a 4 GB storage space, then a 160 GB disk drive can support 40 users. Therefore, a 4096 GB storage system on the server unit can support a total of 1024 the users. Further, any data on a wireless device of a user can be transmitted to an assigned storage volume. In addition, the user of the wireless device also can download multimedia data from an ISP or ASP to the assigned storage volume in a designated server unit through out-band approach shown in FIG. **3**. Finally, for one embodiment, the user can use a web-browser, which has a functionality of invoking embedded video or music, to enjoy his/her stored multimedia contents.

These and other futures, aspects and advantages of the present invention will become understood with reference to the following description, appended claims, and accompanying figures.

DESCRIPTION OF THE DRAWINGS

Referring now to FIG. **1**, which demonstrates an example of an infrastructure comprising a network interconnecting a wireless device and a server. In the FIG. **1**, Net (**2**) represents a communication link, which may be combined with wireless and wired connection media and guarantee that the communication packets can be sent/received between the wireless device and the server. It is also assumed that the net (**2**) representing a communication infrastructure is built up in such way that a user of a wireless device can access and browse any web-site on the Internet, the Intranet.

In the FIG. **1**, the console support software (**5**) on the server (**3**) can be configured to support web-based multi-tasks for a user of the wireless device (**1**) using a web browser **8**. Further, the user of the wireless device is facilitated to perform creating structured layered file directories or folders, and perform data management operations, such as delete, move, copy, rename for data files or folders/directories and etc. on an assigned storage volume configured in the server (**3**).

In addition, the other software modules (**9**) of the wireless device (**1**) is also configured capable to send data to or receive data from the other service modules (**7**) running on the server (**3**) via communication link (**2**) through a suitable IP or non-IP based protocol. The data being sent or received cold be a digital photo picture, a message etc., in respect to a user's request.

Also, the console supporting software (**5**) of the server (**3**) and the other software modules (**9**) of the wireless device (**1**) can be implemented with any suitable languages such as C, C++, Java, etc. depending on the implementation.

Besides, the web-browser (**8**) of the wireless device (**1**) can be any suitable software, which. is capable to communication with web server software (**4**) on the server (**3**) or with any other web server through the HTTP protocol.

On the other hand, FIG. **2** has demonstrated that the storage system (**10**) of a server **3** can be allocated to multiple wireless devices as followings: First, the storage system (**10**) of the server (**3**) can be partitioned into multiple storage volumes (**11**) by administration staff through a web-console (**13**) of a console host (**12**).

Second, the storage system (**10**) of the server (**3**) can be partitioned in such way that a user of each of the wireless devices can be assigned with a storage volume having a desired size, so that the server **3** can support maximum numbers of the wireless devices.

In addition, the storage connection media could be any kind of cables, such as SCSI cable, IP cable, Fiber cable etc. or could be wireless communication media. The storage system itself could be various types.

Finally, the storage system (**10**) can be accessed by each of the wireless devices through IP or non-IP based network and protocols.

FIG. **3** has demonstrated that a user from a web-browser (**8**) on a wireless device (**1**) can download data from a known web-site (**15**) to his/her assigned external storage (**10**) on the server (**3**). The dash-lined path (a) represents a communication channel between the wireless device (**1**) and a remote download web-site (**15**) that provides downloading contents. The dash-lined path (b) represents a communication channel between the wireless devices (**1**) and the storage server (**3**). The dash-lined path (c) represents a communication channel between the server **3** and the remote web-server (**15**).

THE DETAILED DESCRIPTION OF THE INVENTION

The Use of the External Storage by the Wireless Device:

The FIG. **2** shows a simplified diagram of the wireless devices (**1**) using the external storage system (**10**) of [the server (**3**) for effectively resolving the storage limitation problem for the wireless devices (**1**).

Partition Storage Volumes (FIG. **2**)

With this invention, the entire storage (**10**) on the server (**3**) needs to be partitioned into suitable size of volumes (**11**) such as 4 GB for each volume. This will allow the server **3** to serve maximum number of the wireless devices (**1**). With the web console support software (**5**) of the server (**3**), tasks of partitioning the storage system (**10**) can be done through a web-console (**13**) on a console host (**12**) by an administrative staff.

In order to support storage partition, first the console support software (**5**) of the server (**3**) must send storage information of the server (**3**) to the web-console (**13**) of the console host (**12**). The storage information includes each storage device's name and total size etc. Second, based on the received storage information the administration staff on the console host (**12**), for example, can use a web-console (**13**) to partition each storage device and send the storage partition information to the console support software (**5**) of the server (**3**). The storage partition information includes the number of the partitions (volumes) and the size of each partition (volume). Third, upon receiving the storage partition information from the web-console (**13**) of the console host (**12**), the console support software (**5**) of the server (**3**) performs the actual storage partition by dividing the entire storage into multiple small volumes. Finally, for each small storage volume, a corresponding file system could be built up.

Assign Storage Volumes (FIG. **2**)

Each of the storage volumes (**11**) together with its corresponding file system on the storage system (**10**) of the server (**3**) needs to be exclusively assigned to a user of a specific wireless device (**1**) by the console support software (**5**) of the server (**3**).

Data and Storage Volume Management (FIG. **2**)

With the support of the console support software modules (**5**) of the server (**3**), the user of the wireless device (**1**) can utilize the web-browser **8** illustrated in FIG. **2** to setup folder/directory structure on the file system of his/her assigned external storage volume (**11**). In addition, the user of the wireless device (**1**) can use the web-browser **8** performing all data management operations such as delete, copy, move, rename data object etc. on the file system.

In order to support such data management on the external storage (**10**) assigned to the user of the wireless device (**1**) by using the web-browser **8**, first the console support software modules (**5**) of the server (**3**) must communicate with the web-browser (**8**) of the wireless device (**1**). Therefore, the user from the web-browser (**8**) of the wireless device (**1**) can choose desired data management operations and send information of an operation to the console support software modules (**5**) of the server (**3**). The mentioned operation include establishing folder/directory, copying, moving, or reaming data file etc. Second, upon receiving the data management operation, the console support software modules (**5**) of the server (**3**) actually performs these requested operation on the assigned file system of an assigned external storage volume (**11**) on the server (**3**).

Store Data from Wireless Device into External Storage (FIG. **2**)

To store the data such as digital photo pictures or messages into the file system on the assigned storage volume (**11**) in the server (**3**), the other software modules (**9**) of the wireless device (**1**) need to send these data to the other service modules (**7**) of the server (**3**) via communication link between them. Upon receiving the data, the other service modules (**7**) of the server (**3**) write these data to the file system of the assigned storage volume (**11**) for the wireless device **1**. The protocol used between these two communication entities could be either IP or non-IP based protocol.

Case 1:19-cv-01360-RGA Document 1-1 Filed 07/22/19 Page 10 of 12 PageID #: 20

Download Data from a Remote Web Server Site into Allocated Storage Volume:

Now, referring to FIG. **3**, If a user of the wireless device (**1**) wants to download a data from a remote web server (**15**) into the file system on the assigned storage volume (**11**) in the external storage system (**10**) on the server (**3**), the following steps are required:

1) The user of the wireless device (**1**) via a web-browser (**8**) access to a remote web server site (**15**) to obtain information of the data for the downloading via path (a) of FIG. **3**. For example, the user access to a web-page which contains the data name for the downloading.

2) The other software modules (**9**) of the wireless device (**1**) obtain the downloading information for the data, which becomes available in the cached web-pages on the wireless device (**1**) after the web-browser (**8**) access to the web site (**15**).

3) The other software modules (**9**) of the wireless device (**1**) send the obtained downloading information to other service modules (**7**) of the storage server (**3**) via path (b) of FIG. **3**.

4) Upon receiving the downloading information from the wireless device (**1**), the other service module (**7**) of the storage server (**3**) sends a web download request to the web-site (**15**) via path (c) of FIG. **3** based on download information obtained. and receives the downloading data streams from the web server of the web-site (**15**).

5) Upon receiving downloaded data streams, the other service modules (**7**) of the storage server (**3**) write the data streams into the file system of the assigned storage volume (**11**) in the server (**3**) for the wireless device (**1**).

Retrieve Data from Assigned Storage Volume for the User of the Wireless Device

1) If a web-browser (**8**) on a wireless device **1** has embedded video or music functionality, a user of the wireless device (**1**) can use the browser to retrieve and play multimedia data file such as video or music stored in the assigned storage volume (**11**) located on the server (**3**).

2) In another embodiment, in respect to the user's needs, the other software module (**9**) of the wireless device (**1**) also can retrieve data file from the file system of the assigned storage volume (**11**) on the server (**3**).

Support External Storage for a Large Number of the Wireless Devices

If there is a need to provide each user a 2 GB of storage space, then a 160 GB disk drive can support 80 users. A 4096 GB (4 Tera Bytes) storage system on the server unit can support 2024 user. Each of the server units only can efficiently support a limited size of the storage system. In order to support a large number of the wireless devices, such as for 500,000 wireless devices, a larger number of the servers is required, in this case 250 servers is required. In order to let a larger number of the servers to effectively support the larger number of the wireless devices, an infrastructure like CCDSVM is desirable, which has been described in prior patent applications. With the CCDSVM the control management station can control larger number of storage servers to provide external storage for a huge number of the wireless devices.

The invention claimed is:

1. A wireless device comprising:

at least one cache storage, one wireless interface, and program code configured to cause the wireless device to:

establish a wireless link for the wireless device access to a storage space of a predefined capacity assigned exclusively to a user of the wireless device by a storage server, and

couple with the storage server across the wireless link to carry out a requested operation for remote access to the assigned storage space in response to the user from the wireless device performed the operation,

wherein the operation for the remote access to the assigned storage space comprises storing a data object therein or retrieving a data object therefrom, the storing of a data object including to download a file from a remote server across a network into the assigned storage space through utilizing download information for the file stored in said cache storage in response to the user from the wireless device performed the operation for downloading the file from the remote server into the assigned storage space.

2. The wireless device as recited in claim **1**, wherein the data object, being stored into or retrieved from the assigned storage space, comprises a message or multimedia data of video, digital music, or digital picture.

3. The wireless device as recited in claim **1**, wherein the storage server controls a plurality of storage devices, one of the storage devices being configured with the storage space assigned exclusively to the user.

4. The wireless device as recited in claim **1**, wherein said downloading a file from a remote server further comprises:

obtaining downloading information for the file;

transmitting the downloading information cached in the wireless device to the storage server; and

causing the storage server in accordance with the downloading information to download the file into the assigned storage space.

5. The wireless device as recited in claim **1**, wherein the wireless device further is one of a cell phone or a personal data assistant and management device ("PDA").

6. The wireless device as recited in claim **1**, wherein said operation for remotely access to the assigned storage space further comprises:

from the wireless device creating a folder structure in the assigned storage space.

7. The wireless device as recited in claim **1**, wherein said operation for remotely access to the assigned storage space further comprises:

from the wireless device deleting or moving or copying or renaming a folder in the assigned storage space.

8. The wireless device as recited in claim **6**, wherein said operation for remotely access to the assigned storage space further comprises:

from the wireless device deleting or moving or copying or renaming a file in the assigned storage space.

9. The wireless device as recited in claim **1**, wherein wireless device further executes a web browser for the user access to the assigned storage space, access to Internet.

10. The wireless device as recited in claim **6**, wherein said operation for remote access the assigned storage space further comprises:

from the wireless device creating a folder in the assigned storage space.

11. A non-transitory computer-readable medium comprising program code that, being executed by a wireless device, causes the wireless device to:

establish a wireless link for the wireless device access to a storage space of predefined capacity assigned exclusively by a storage server to a user of the wireless device;

couple with the storage server through the wireless link to carry out a requested operation for remote access to the assigned storage space in response to the user from the wireless device performed the operation, wherein the operation for the remote access to the assigned storage space comprises storing a data object therein or retrieving a data object therefrom, the storing of the data object including to download a file from a remote server on a network into the assigned storage space through utilizing download information for the file stored in a cache storage of the wireless device in response to the user from the wireless device performed the operation for downloading the file from the remote server into the assigned storage space.

12. The non-transitory computer-readable medium as recited in claim **11**, wherein the program code causes the wireless device, in response to said downloading of a file, obtaining downloading information for the file, transmitting the downloading information cached in the wireless device to the storage server, and causing the storage server in accordance with the downloading information to download the file into the assigned storage space.

13. The non-transitory computer-readable medium as recited in claim **11**, wherein the program further causes the wireless device coupling with the storage server to carry out an operation for remotely deleting, moving, copying, or renaming a folder in the assigned storage space.

14. The non-transitory computer-readable medium as recited in claim **11**, wherein the program code further causes the wireless device coupling with the storage server to carry out an operation for remotely creating a folder structure in the assigned storage space.

15. The non-transitory computer-readable medium as recited in claim **14**, wherein the program code further causes the wireless device coupling with the storage server to carry out an operation for remotely creating a folder in the assigned storage space.

16. The non-transitory computer-readable medium as recited in claim **11**, wherein said data object being stored in or retrieved from the assigned storage space is one of a message or multimedia data of video, digital music or digital picture.

17. The non-transitory computer-readable medium as recited in claim **13**, wherein the program code further causes the wireless device coupling with the storage server to carry out an operation for remotely deleting, moving, copying, or renaming a file in the assigned storage space.

18. The non-transitory computer-readable medium as recited in claim **11**, wherein the program code further causes the wireless device executing a web browser through which the user access to the assigned storage space, and access to Internet.

19. The non-transitory computer-readable medium as recited in claim **11**, wherein the wireless device further is one of a cell phone or a personal data assistant and management device ("PDA").

20. The non-transitory computer-readable medium as recited in claim **11**, wherein the storage server controls a plurality of storage devices, one of the storage devices configured with the storage space assigned exclusively to the user.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 9,098,526 B1
APPLICATION NO. : 14/150106
DATED : August 4, 2015
INVENTOR(S) : Tsao

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

In the Specification
1) In col. 2, line 42, please replace the “volume” with --- volumes ---;
2) In col. 3, line 65, please replace “[the” with --- the ---;

In the Claims
3) In col. 6, line 52, please replace the “Internet” with --- the Internet ---;
4) In col. 7, line 5, please replace the “on a” with --- across a ---;
5) In col. 7, line 21, please replace the “program” with --- program code ---;
6) In col. 6, line 4, line 12, and line 67, and col. 7, line 8, please replace “performed” with --- performing ---.

Signed and Sealed this
Twenty-ninth Day of March, 2016

Michelle K. Lee
Director of the United States Patent and Trademark Office

EXHIBIT B

US010015254B1

(12) **United States Patent**
Tsao

(10) **Patent No.: US 10,015,254 B1**
(45) **Date of Patent: *Jul. 3, 2018**

(54) **SYSTEM AND METHOD FOR WIRELESS DEVICE ACCESS TO EXTERNAL STORAGE**

(71) Applicant: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(72) Inventor: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(73) Assignee: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **14/977,509**

(22) Filed: **Dec. 21, 2015**

Related U.S. Application Data

(63) Continuation of application No. 14/036,744, filed on Sep. 25, 2013, now Pat. No. 9,239,686, which is a continuation of application No. 10/726,897, filed on Dec. 4, 2003, now Pat. No. 8,606,880.

(51) **Int. Cl.**

G06F 3/00	(2006.01)
H04L 29/08	(2006.01)
G06F 3/06	(2006.01)
H04M 1/725	(2006.01)

(52) **U.S. Cl.**
CPC ***H04L 67/1097*** (2013.01); ***G06F 3/0604*** (2013.01); ***G06F 3/067*** (2013.01); ***G06F 3/0643*** (2013.01); ***H04L 67/06*** (2013.01); ***H04M 1/72522*** (2013.01)

(58) **Field of Classification Search**
CPC H04L 67/1097; H04L 67/06; H04M 1/72522; G06F 3/0604; G06F 3/0643; G06F 3/067
USPC 711/118, 154, 171; 709/219, 226; 710/36; 712/225, 245
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

6,292,833 B1 * 9/2001 Liao H04W 12/08 709/219
6,351,776 B1 * 2/2002 O'Brien G06F 17/30067 707/999.001
6,356,838 B1 * 3/2002 Paul G01C 21/3453 701/410
6,754,710 B1 * 6/2004 McAlear G08C 17/00 709/208
6,757,898 B1 * 6/2004 Ilsen G06F 19/324 600/300
7,379,990 B2 * 5/2008 Tsao H04L 41/12 709/220
7,418,702 B2 * 8/2008 Tsao G06F 9/526 709/229

(Continued)

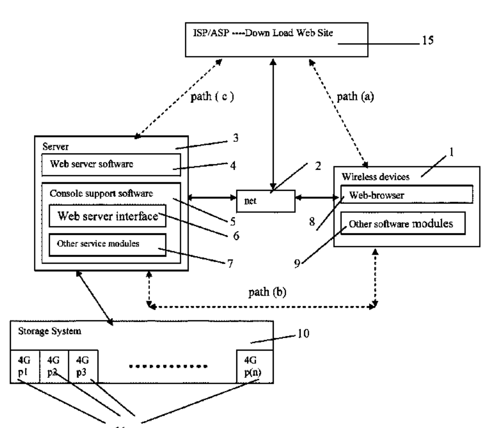
Primary Examiner — Reba I Elmore

(57) **ABSTRACT**

Traditionally, wireless device, such as cell phone or personal data assistant device (PDA), has relatively smaller storage capacity. Therefore, it is quite often that a user of the wireless device has difficulty to find more storage space for storing ever increased personal data, such as storing message, and multiple Gig bytes of multimedia data including digital video, music, or photo picture etc. Instant application disclosed a system and method for a storage system providing storage service to the wireless device for the wireless device remotely storing personal data into an external storage space allocated exclusively to a user of the wireless device by the storage system.

20 Claims, 4 Drawing Sheets

Wireless out-band download



(56) **References Cited**

U.S. PATENT DOCUMENTS

8,606,880	B2 *	12/2013	Tsao	G06F 17/30067 455/412.1
9,098,526	B1 *	8/2015	Tsao	G06F 17/30067
9,219,780	B1 *	12/2015	Tsao	G06F 17/30067
9,239,686	B2 *	1/2016	Tsao	G06F 3/0631
2002/0004855	A1 *	1/2002	Cox	G06Q 30/02 719/328
2003/0018887	A1 *	1/2003	Fishman	H04L 29/06 713/151
2003/0046586	A1 *	3/2003	Bheemarasetti	H04L 63/0272 726/15
2003/0078986	A1 *	4/2003	Ayres	H04L 29/06 709/217
2003/0194998	A1 *	10/2003	Bhide	H04W 4/18 455/433
2003/0233329	A1 *	12/2003	Laraki	G06Q 20/123 705/52
2004/0162876	A1 *	8/2004	Kohavi	G06F 17/30943 709/203
2004/0192342	A1 *	9/2004	Ranganathan	H04L 65/4092 455/456.1
2005/0010648	A1 *	1/2005	Ramaswamy	H04L 12/5692 709/217
2005/0071418	A1 *	3/2005	Kjellberg	G06Q 20/123 709/200
2005/0091184	A1 *	4/2005	Seshadri	G06F 9/4443
2007/0089110	A1 *	4/2007	Li	H04L 67/2847 717/178
2008/0235360	A1 *	9/2008	Li	G06F 17/30902 709/223

* cited by examiner

Wireless devices supports in a simple environment

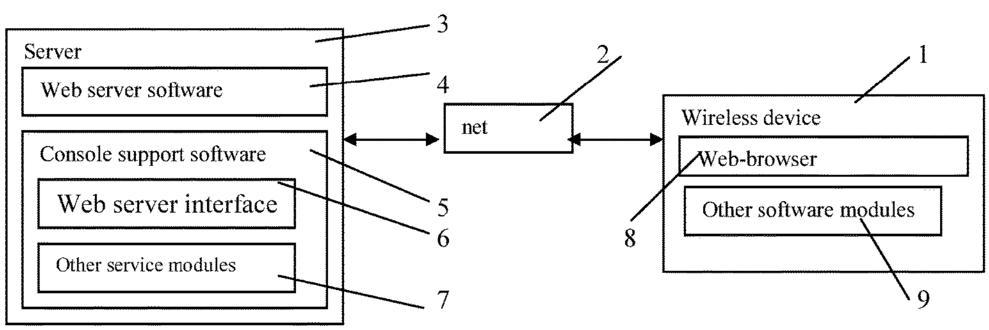


Fig. 1 (Prior Art)

Wireless devices access external storage through web browser

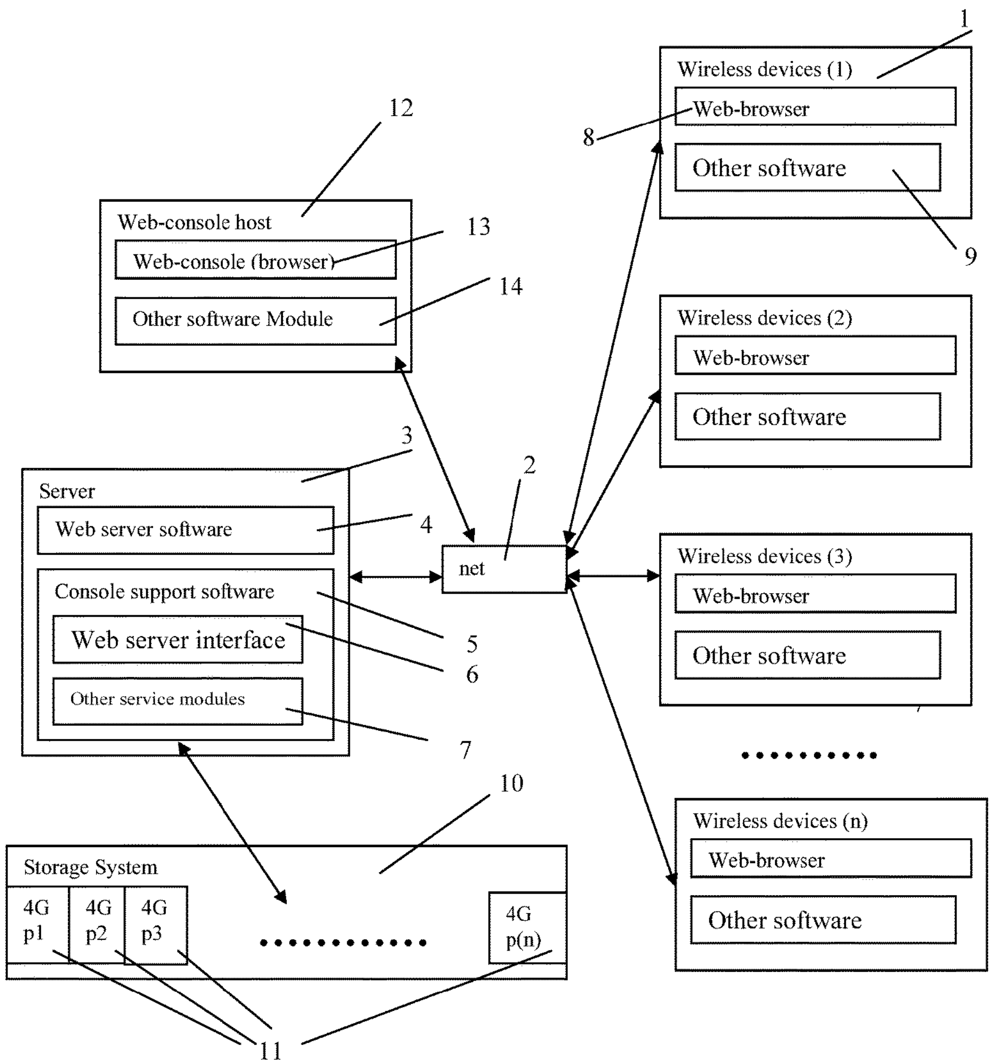


Fig. 2

Wireless out-band download

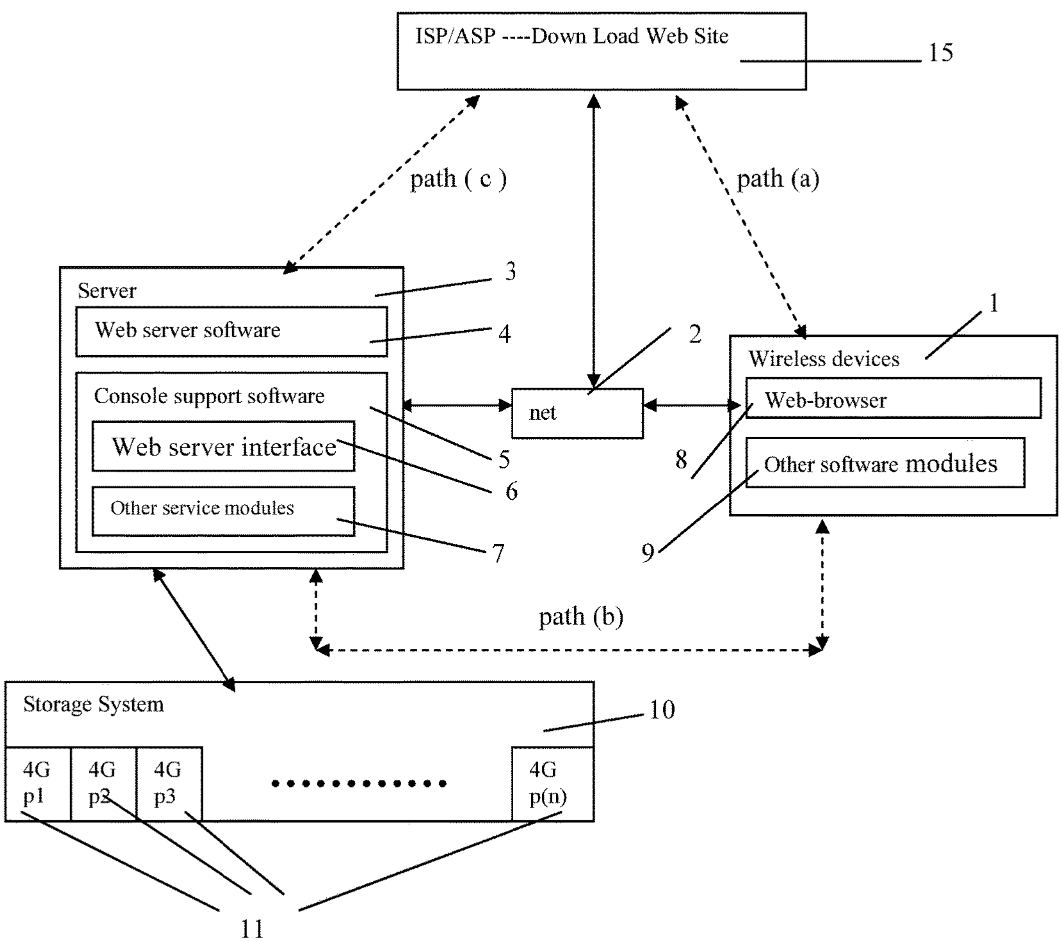


Fig. 3

The CCDSVM Support External Device for Huge Number of Wireless Device

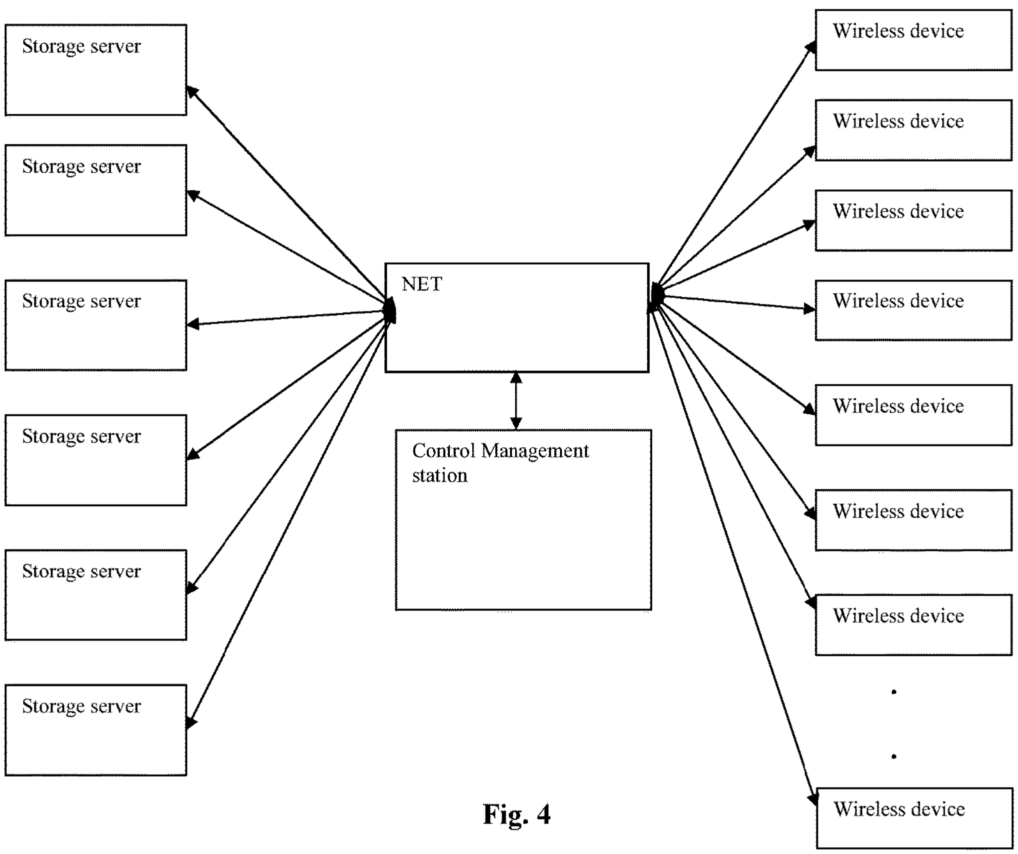


Fig. 4

SYSTEM AND METHOD FOR WIRELESS DEVICE ACCESS TO EXTERNAL STORAGE

CROSS REFERENCE TO PRIOR APPLICATION

This invention is a continuation application of the U.S. patent application Ser. No. 14/036,744 filed on Sep. 25, 2013, now a U.S. Pat. No. 9,239,686 and which itself is continuation application for U.S. patent application Ser. No. 10/726,897 filed on Dec. 4, 2003 and now a U.S. Pat. No. 8,606,880. The application Ser. No. 10/726,897 has also referenced application Ser. No. 10/713,904 of "Concurrent Web Based Multi-task Support for Control Management System" and application Ser. No. 10/713,905 of "IP Based Distributed Virtual SAN" in the name of same inventor. All mentioned prior applications and patents are herein incorporated by reference in their entirety for all purpose.

FIELD OF THE INVENTION

This invention relates to wireless devices accessing and using external storage spaces provided by one or more servers.

BACKGROUND INFORMATION

Storage system can be categorized as internal storage or external storage system.

The internal storages of a computing system include those storage media such as hard disk drives, memory sticks, memory, and others that are internally connected within the computing system through system bus or a few inches of cable. Therefore, the storage media actually are internal components of the computing system in a same enclosure.

The external storages of a computing system are those storage media that are not the internal components of the computing system in a same enclosure. Therefore, they have to be connected through longer cable, such as through Ethernet cable for IP based storage, Fiber channel cable for fiber channel storage, or wireless communication media, and others. The storage media of the external storage could be magnetic hard disk drives, solid state disk, optical storage drives, memory card and others, and could be in any form such as Raid which usually consists of a group of hard disk drives.

To effectively use a storage system, the storage devices of the storage system usually need to be partitioned into small volumes. After the partitioning, each of the volumes can be used for establishing a file system on top of it. To simplify the discussion, herein, the term of the storage volume and its corresponding file system, and the term of the storage partition are often used without differentiation in this invention.

CCDSVM in this invention is an abbreviation for a central controlled distributed scalable virtual machine. The CCDSVM allows a control management system to control a group of computing systems for providing distributed services to client systems over the Internet, Intranet, and LAN environment.

By the way, in this invention, the ISP refers to Internet service provider and the ASP refers to application service provider.

BRIEF DESCRIPTION OF FIGURES

FIG. **1** is a figure the same as a FIG. **1** of a prior application Ser. No. 10/713,904 of the "Concurrent Web Based Multi-task Support for Control Management System" with exceptions that a console host of the prior application being replaced herein by a wireless device.

FIG. **2** is a figure the same as the FIG. **1** of this invention with an exception that it shows a more detailed storage system configured and controlled by the server in the FIG. **1** above. In addition, multiple wireless devices are presented in this FIG. **2**.

FIG. **3** shows a scheme of a wireless device downloading data (contents) from an ISP/ASP or from other web sites to an external storage of the wireless device.

FIG. **4** is a figure the same as a FIG. **1** of the prior application Ser. No. 10/713,905 of "IP Based Distributed Virtual SAN" with an exception that each IP storage server herein provides file system for external storage to each of the wireless devices instead of providing IP based virtual SAN service of the prior application. Also, each host in the FIG. **1** of the prior application has been replaced by a wireless device of this invention.

Unless specified, the programming languages and the protocols used by the software modules, and the computing systems used in this invention are assumed to be the same as described in the prior patent applications.

In addition, in the drawing, like elements are designated by like reference numbers. Further, when a list of identical elements is present, only one element may be given the reference number.

BRIEF DESCRIPTION OF THE INVENTION

Today, users commonly face a problem of lack of storage capacity in their wireless devices such as in their cell phones or personal data assistant devices ("PDA"), which are usually limited to 256 MB for the PDA and much less for the cell phone. To effectively solve this problem and let users possess multiple gigabytes (GB) of storage for their wireless devices as well as allowing the users to use the GB storage for their multimedia applications, the storage on a server can be used as the external storage for the wireless devices. This technology has been briefly introduced in the prior patent applications by the same author.

The followings describe details on how can the external storage actually be used by the wireless devices by referring to the FIG. **2**. First, let each server unit (e.g. the server **3** of the FIG. **2**) partition its storage system in such way that each of volumes will have multiple GB in size. Therefore, a user of any one of the wireless devices can exclusively be assigned for access to a specific storage volume on the server unit **3**. For example, if we need to provide each user a 4 GB storage space, then a 160 GB disk drive can support 40 users. Therefore, a 4096 GB storage system on the server unit **3** can support a total of 1024 users. Further, any data on the wireless device can be transmitted to the assigned storage volume on the server unit. In addition, the user on the wireless device also can download multimedia data from an ISP or ASP to the assigned storage volume of the designated server unit **3** through out-band approach as shown in FIG. **3**. Finally, the user can use a web-browser, which has a functionality of invoking embedded video or music, to enjoy his/her stored multimedia contents.

These and other features, aspects and advantages of the present invention will become understood with reference to the following description, appended claims, and accompanying figures.

DETAILED DESCRIPTION OF THE DRAWINGS

Referring now to the FIG. **1**, this figure demonstrates a network connection between a wireless device and a server, where Net (**2**) represents a communication link, which may be combined with wireless and wired connection media and guarantee that the communication packets can be sent or received between the wireless device and the server. It is also assumed that the net (**2**) infrastructure is built up in such way that a user from a web-browser of a wireless device can access and browse any web-site on the Internet, and Intranet.

In the FIG. **1**, the console support software (**5**) on the server (**3**) can support web-based multitasking while a user using a web-browser (**8**) of a wireless device (**1**). Further, the user on the web-browser (**8**) is facilitated to perform creating structured layered file directories or folders, and perform data management operations, such as delete, move, copy, rename for data files or folders or directories residing on an assigned storage volume of the server (**3**).

In addition, the other software modules (**9**) of the wireless device (**1**) is also configured capable to send data to or receive data from the other service modules (**7**) of the server (**3**) via communication link (**2**) through a suitable IP or non-IP based protocol. The data being sent or received could be a digital photo picture, a message and other data.

In addition, the console supporting software (**5**) of the server (**3**) and the other software modules (**9**) of the wireless device (**1**) can be implemented with suitable languages such as C, C++, Java, and others.

Besides, the web-browser (**8**) of the wireless device (**1**) can be any suitable software tool, which. is capable to communication with web server software (**4**) on the server (**3**) or with other web server through the HTTP protocol.

The FIG. **2** has demonstrated how can the storages of a server (**3**) be assigned to multiple wireless devices for being used as their external storage as follows:

First, the storage system (**10**) of the server (**3**) can be partitioned into multiple volumes (**11**), for example, by administration staff through a web-console (**13**) of a console host (**12**).

Second, the storage system (**10**) of the server (**3**) can be partitioned in such way that each of the wireless devices can be assigned with a storage volume of a desired size, which can be best supported by the server (**3**) for supporting maximum numbers of the wireless devices.

In addition, the storage connection media could be any kind such as SCSI cable, IP cable, Fiber cable and others. Meanwhile, the storage system itself could be various types.

Finally, the storage system (**10**) can be accessed by each of the wireless devices through IP or non-IP based network and protocols.

The FIG. **3** has demonstrated that a user from a web-browser (**8**) on a wireless device (**1**) can download data from a known web-site (**15**) to his/her assigned external storage (**10**) on the server (**3**). The dash-lined path (a) represents a communication link between the wireless device (**1**) and the remote web-site (**15**), which provides contents for web download. The dash-lined path (b) represents a communication link between the wireless devices (**1**) and the storage server (**3**). The dash-lined path (c) represents a communication link between the server **3** and the remote web-server (**15**), which provides download contents.

THE DETAILED DESCRIPTION OF THE INVENTION

The Use of the External Storage of the Wireless Device:

The FIG. **2** shows a simplified diagram illustrating the wireless devices (**1**) using and accessing the external storage system (**10**) of a server (**3**) for solving the storage limitation problem of the wireless devices (**1**).

Partition Storage Volumes (Refer to the FIG. **2**):

With this invention, the entire storage (**10**) on a server (**3**) needs to be partitioned into suitable size of volumes (**11**) such as 4 GB for each volume for allowing the server **3** to serve maximum number of the wireless devices (**1**). With the web console support software (**5**) of the server (**3**), the task of partitioning the storage system (**10**) can be done through a web-console (**13**) on a console host (**12**) by an administrative staff.

In order to support storage partition, first the console support software (**5**) of the server (**3**) must send storage information of the server (**3**) to the web-console (**13**) of the console host (**12**). The storage information includes each storage device name, storage size and others. Second, based on the storage information received, the administration staff on the console host (**12**) can use the web-console (**13**) to fill (partition each storage device) and send the storage partition information to the console support software (**5**) of the server (**3**). The storage partition information includes the number of the partitions (volumes) and the size of each partition (volume). Third, upon receiving the storage partition information from the web-console (**13**) of the console host (**12**), the console support software (**5**) of the server (**3**) performs the actual storage partition by dividing the entire storage into multiple small volumes. Finally, for each small storage volume, a corresponding file system could be built up.

Assign Storage Volumes (Refer to the FIG. **2**):

Each of the storage volumes (**11**) together with its corresponding file system (**11**) on the storage system (**10**) of the server (**3**) needs to be exclusively assigned and exported to a given specific wireless device (**1**) by the console support software (**5**) of the server (**3**).

Data and Storage Volume Management (Refer to FIG. **2**)

With the support of the console support software modules (**5**) of the server system-(**3**), a user on a web-browser (**8**) of the wireless device (**1**) can setup folder or directory structure on the file system of his/her assigned external storage volume (**11**). In addition, the user on the web-browser (**8**) of the wireless device (**1**) can performing all data management operations such as delete, copy, move, or rename and other operations for files or folders on that file system.

In order to support such data management over the external storage (**10**) by using the web-browser (**8**) of the wireless device (**1**), first the console support software modules (**5**) of the server system (**3**) must communicate with the web-browser (**8**) of the wireless device (**1**) for presenting the external storage to a user on a user device. Therefore, the user from the web-browser (**8**) of the wireless device (**1**) can choose and submit desired data management operation for the wireless device (**1**) sending the operation information to the console support software modules (**5**) of the server system (**3**). These operations include establishing folder or directory, and copying, moving, or renaming data file and others for the folder or directory. Second, upon receiving each data management operation, the console support software modules (**5**) of the server system (**3**) actually performs the operation over the assigned file system of the assigned external storage volume (**11**) on the server system (**3**).

Store Data from Wireless Device into External Storage (Refer to FIG. **2**):

To store the data such as digital photo pictures or messages into the assigned file system on the external storage (**11**) of a server (**3**), the other software modules (**9**) of the wireless device (**1**) need to send the data to the other service modules (**7**) of the server (**3**) via communication link between them. Upon receiving the data, the other service modules (**7**) of the server (**3**) write these data to the assigned

Case 1:19-cv-01360-RGA Document 1-3 Filed 07/22/19 Page 10 of 11 PageID #: 40

file system of the assigned storage volume (**11**) on the server (**3**). The protocol used between these two communication entities could be either IP or non-IP based protocol.

Download Data from a Remote Web Server Site into External Storage (Refer to the FIG. **3**):

If a user of a wireless device (**1**) wants to download data from a remote web server (**15**) into an assigned file system (**11**) of the assigned external storage on a server (**3**), the following steps are required:

1) Provide the user from a web-browser (**8**) of the wireless device (**1**) access to a remote web server site (**15**) to obtain information for the downloading via the path (a) of the FIG. **3**. For example, to provide the user via the web browser obtains a web-page, which contains IP address of the remote web site and the data name for the downloading.

2) The other software modules (**9**) of the wireless device (**1**) obtain the downloading information, which becomes available in the cached web-pages on the wireless device (**1**) after the web-browser (**8**) accessing the web site (**15**).

3) The other software modules (**9**) of the wireless device (**1**) send the obtained downloading information to other service modules (**7**) of the storage server (**3**) via the path (b).

4) Upon receiving the downloading information from the wireless device (**1**), the other service module (**7**) of the storage server (**3**) sends a web download request to the web-site (**15**) via the path (c) based on download information obtained and then receives the downloading data from the web server of the web-site (**15**).

5) Upon receiving downloading data, the other service modules (**7**) of the storage server (**3**) write the data for the wireless device (**1**) into the assigned file system (**11**) on the server (**3**).

Retrieve Data from External Storage of a Wireless Device:

1) If a web browser (**8**) has embedded video or music functionality, the web-browser (**8**) of a wireless device (**1**) can be used to retrieve and play multimedia data file such as video or music stored in the wireless device's external storage volume (**11**), which actually located on a server (**3**).

2) If there is needs, the other software module (**9**) of the wireless device (**1**) also can retrieve data file from the assigned file system of the assigned storage volume (**11**) on a server (**3**).

Support External Storage for Large Number of the Wireless Devices:

Referring now to the FIG. **4**. If there is a need to provide each user a 2 GB of storage space, then a 160 GB disk drive can support 80 users. A 4096 GB (4 Tera Bytes) storage system on the server unit can support 2024 user. Therefore, a server unit **3** only can efficiently support a limited size of the storage system. In order to support a large number of the wireless devices, such as for supporting 500,000 wireless devices, a larger number of the servers is required, in this case **250** servers is required. In order to let a larger number of the servers to effectively support the larger number of the wireless devices, an infrastructure like the CCDSVM is desirable, which has been described in the prior patent applications. With the CCDSVM the control management system can control larger number of storage servers to provide external storage for a huge number of the wireless devices.

What is claimed is:

1. A wireless device accessing a remote storage space, the wireless device comprising:

at least one cache storage for caching data received from the Internet, and

one computer-readable storage device comprising program instructions which, when executed by the wireless device, configure the wireless device accessing the remote storage space, wherein the program instructions comprise:

program instructions for the wireless device establishing a communication link for accessing the remote storage space served by a first server;

program instructions for the wireless device displaying the remote storage space upon receiving information of the remote storage space from the first server; and

program instructions for the wireless device coupling with the first server to carry out a requested operation for accessing the remote storage space in response to a user, through the remote storage space displayed on the wireless device, performing the operation,

wherein the operation being carried out for accessing the remote storage space comprises from the wireless device storing data therein or retrieving data therefrom, the storing data comprising to download a file from a second server across a network into the remote storage space through utilizing information for the file cached in the cache storage in the wireless device.

2. The wireless device of claim **1**, wherein said downloading a file from a second server comprises:

program instructions for the wireless device obtaining the information for the file from the second server, and

transmitting the information for the file cached in the wireless device to the first server to cause the first server, in accordance with the information for the file, to download the file from the second server into the remote storage space.

3. The wireless device of claim **1**, wherein said information for the file cached in the wireless device comprises at least the name of the file and the internet protocol ("IP") address of the second server.

4. The wireless device of claim **3**, wherein said displaying the remote storage space further comprises program instruction for the wireless device displaying information of the remote storage space in a web browser on the wireless device for accessing the remote storage space or accessing the Internet.

5. The wireless device of claim **1**, wherein the wireless device is one of a cell phone, or a personal data assistant and management device ("PDA").

6. The wireless device of claim **1**, wherein said operation for accessing the remote storage space comprises:

from the wireless device and via a wireless link creating a folder, or a layered folder or directory structure in the remote storage space.

7. The wireless device of claim **6**, wherein said operation for access the remote storage space comprises:

from the wireless device moving, copying, deleting or renaming a data object in respect to the layered folder or directory structure, wherein said data object is a file or folder.

8. The wireless device of claim **1**, wherein said operation for accessing the remote storage space comprises: from the wireless device and via a wireless link storing therein or retrieving therefrom a data object of a message, or a digital video, music, or picture file.

9. A server for delivering storage service, comprising:

a plurality of storage spaces residing among a plurality of storage devices; and

a computer-readable storage device comprising program instructions that, when executed by the server, config-

Case 1:19-cv-01360-RGA Document 1-3 Filed 07/22/19 Page 11 of 11 PageID #: 41

ure the server to control delivering the storage service; wherein the program instructions comprise:

program instructions for the server establishing a communication link for a first wireless device remotely accessing a first one of the storage spaces;

program instructions for the server sending information of the first one of the storage spaces to the first wireless device for causing display of the information on the first wireless device; and

program instructions for the server updating the first one of the storage spaces according to a requested operation received from the first wireless device upon a user thereof, through the displayed information of the first one of the storage spaces performing the operation for remotely accessing the first one of the storage spaces,

wherein said operation for remotely accessing the first one of the storage spaces comprises from the first wireless device storing data therein or retrieving data therefrom, wherein the storing data further comprises program instructions for the server downloading a file from a remote server across a network into the first one of the storage spaces through utilizing information for the file cached in a cache storage in the first wireless device.

10. The server of claim **9**, wherein said downloading a file through utilizing information for the file cached in the first wireless device comprises program instructions for the server receiving the information for the file from the first wireless device, and sending a request of downloading the file to the remote server according to the information for the file received.

11. The server of claim **9**, wherein the server presents a second one of the storage spaces to a user on a second wireless device for facilitating the user to perform operations for remotely accessing the second one of the storage spaces.

12. The server of claim **9**, wherein said causing display of the first one of the storage spaces further comprises: causing display of the first one of the storage spaces in a web browser on the first wireless device for accessing the first one of the storage spaces.

13. The server of claim **9**, wherein said program instructions configuring the server to control delivering the storage service comprise program instructions for the server to send information of said plurality of storage devices to a web console for partition each of said storage devices for creating the storage spaces.

14. The server of claim **9**, wherein said operation further comprises: from the first wireless device remotely creating a folder or a layered folder or directory structure in the first one of the storage spaces, or remotely deleting, moving, copying or renaming a data object in respect to the layered folder structure, wherein the data object is a file, a folder, where the file includes digital picture, music or video file.

15. The server of claim **9**, wherein said operation for accessing the first one of the storage spaces comprises: from the wireless device and via a wireless link storing therein or retrieving therefrom a data object of message, digital video, digital music, or digital picture.

16. A method for a wireless device accessing a remote storage space, the method comprising actions performed by the wireless device, including:

establishing a communication link for accessing the remote storage space served by a first server;

displaying the remote storage space on the wireless device upon receiving information of the remote storage space from the first server; and

coupling with the first server to carry out a requested operation for accessing the remote storage space in response to a user, through the remote storage space displayed on the wireless device, performing the operation,

wherein the operation being carried out for accessing the remote storage space comprises from the wireless device storing data therein or retrieving data therefrom, the storing data comprising to download a file from a second server across a network into the remote storage space through utilizing information for the file cached in a cache storage in the wireless device.

17. The method of claim **16**, wherein said downloading a file from a second server comprises:

the wireless device obtaining the information for the file from the second server,

transmitting the information for the file cached in the cache storage in the wireless device to the first server, and

causing the first server in accordance with the received information for the file to download the file from the second server into the remote storage space.

18. The method of claim **16**, wherein said wireless device displaying the remote storage space further comprises displaying the remote storage space in a web browser for accessing the remote storage space.

19. The method of claim **16**, wherein said operation for accessing the remote storage space comprises:

from the wireless device and via a wireless link creating a folder, or a layered folder or director structure in the remote storage space.

20. The method of claim **19**, wherein said operation for accessing the remote storage space comprises:

from the wireless device and via a wireless link moving, copying, deleting, or renaming a data object of file or folder in respect to the layered folder or director structure.

* * * * *

EXHIBIT C

US008606880B2

(12) **United States Patent**
Tsao

(10) **Patent No.: US 8,606,880 B2**
(45) **Date of Patent: Dec. 10, 2013**

(54) **USE OF WIRELESS DEVICES' EXTERNAL STORAGE**

(75) Inventor: **Sheng (Ted) Tai Tsao**, San Jose, CA (US)

(73) Assignee: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 2766 days.

(21) Appl. No.: **10/726,897**

(22) Filed: **Dec. 4, 2003**

(65) **Prior Publication Data**

US 2010/0005153 A1 Jan. 7, 2010

(51) **Int. Cl.**
G06F 15/16 (2006.01)

(52) **U.S. Cl.**
USPC **709/219**; 709/203; 709/226; 455/412.1; 455/899

(58) **Field of Classification Search**
USPC 709/200, 203, 217, 219, 226, 245; 455/412.1, 899
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

6,351,776	B1 *	2/2002	O'Brien et al.	709/245
6,356,941	B1 *	3/2002	Cohen	709/219
6,714,968	B1 *	3/2004	Prust	709/219
6,735,623	B1 *	5/2004	Prust	709/219
6,839,743	B2 *	1/2005	Shim	709/217
6,985,927	B2 *	1/2006	O'Brien et al.	709/213
7,506,034	B2 *	3/2009	Coates et al.	709/219
7,650,621	B2 *	1/2010	Thomas et al.	725/87
7,894,803	B2 *	2/2011	Kamada	455/414.3
2002/0059621	A1 *	5/2002	Thomas et al.	725/87
2002/0123336	A1 *	9/2002	Kamada	455/420
2002/0128036	A1 *	9/2002	Yach et al.	455/552
2002/0133597	A1 *	9/2002	Jhingan et al.	709/228
2003/0191716	A1 *	10/2003	Woods et al.	705/50

* cited by examiner

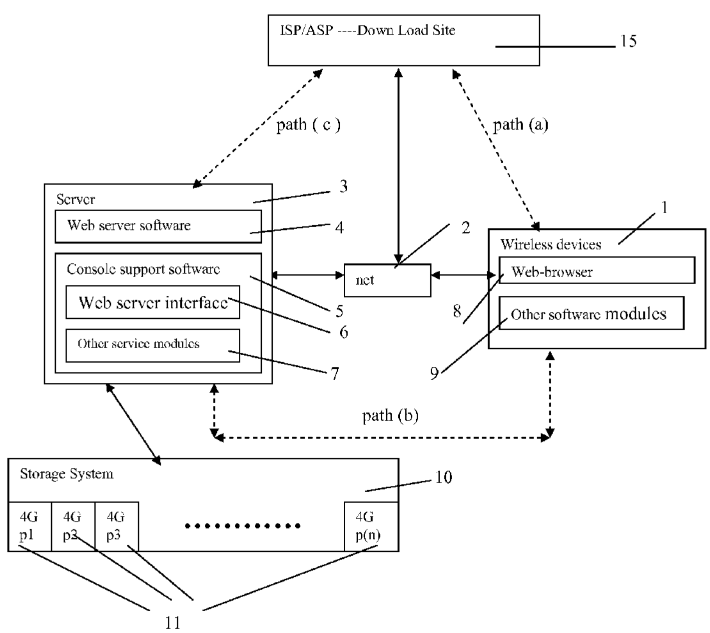
Primary Examiner — El Hadji Sall

(57) **ABSTRACT**

Adapting web-based external storage, wireless device can posses huge amount of storage that current any wireless device's internal storage can not provide. To effectively let the storage server providing external storage (file system) for wireless device, the storage of a storage server need to be partitioned into multiple small storage volume and need to be exported to each specific wireless device. The console support software coupled with web-server software of a server provides both users of wireless device and console through web-browser to perform tasks of creating and utilizing external storage (file system). To support larger number of wireless devices with external storage, a central controlled distributed scalable virtual machine infrastructure can be deployed. The larger number of storage server controlled by a central control system can satisfy unlimited wireless devices external storage needs.

17 Claims, 18 Drawing Sheets

Wireless out-band download



Wireless devices supports in a simple environment

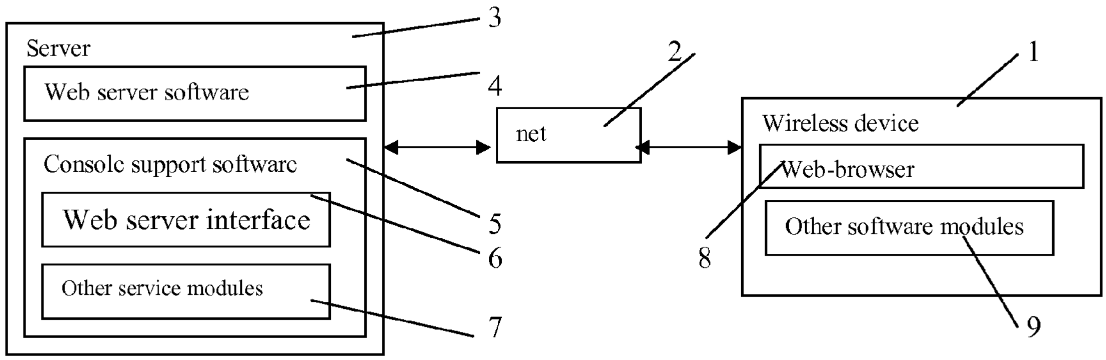


Fig. 1

Wireless devices access external storage through web browser

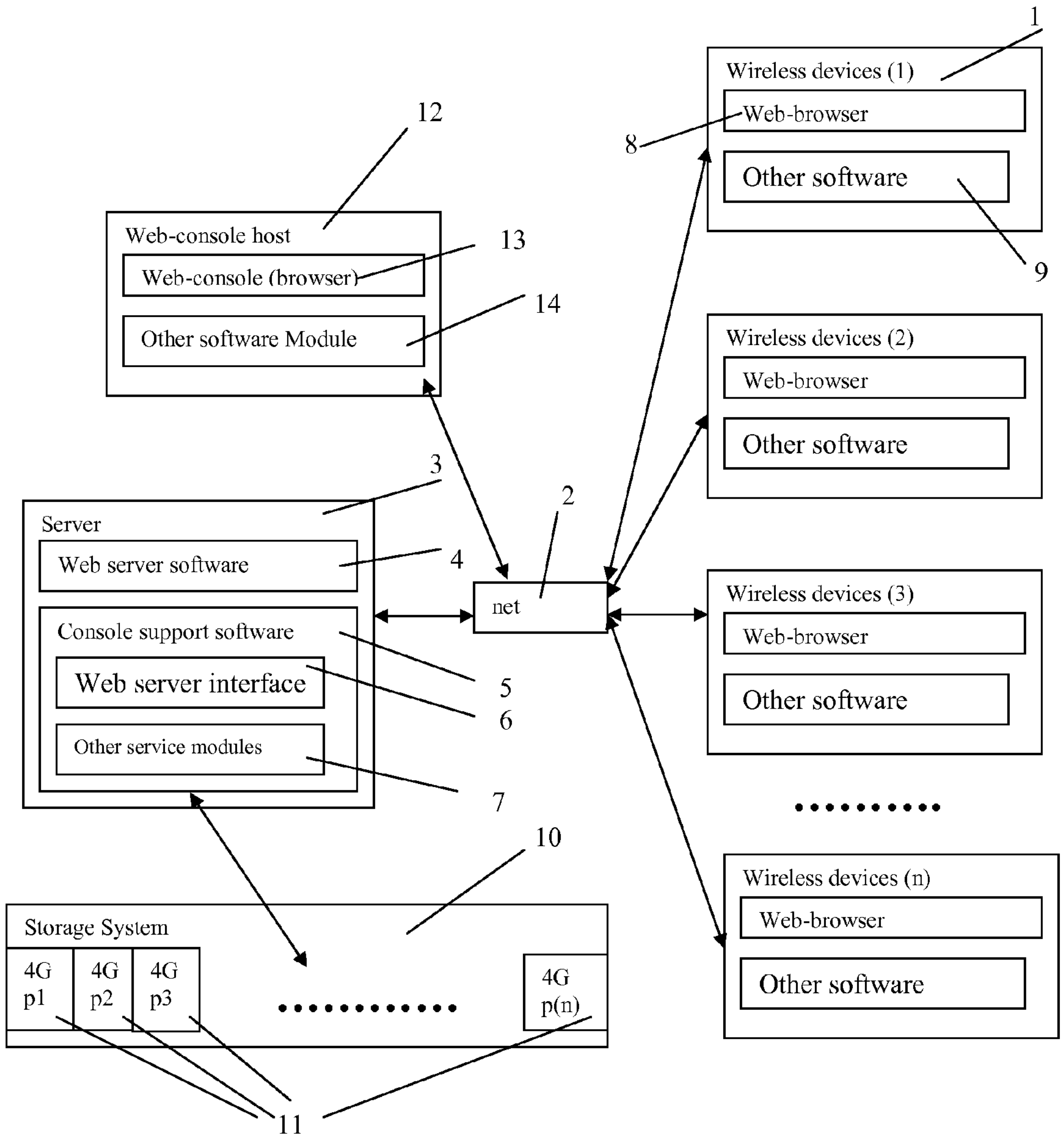


Fig. 2

Wireless out-band download

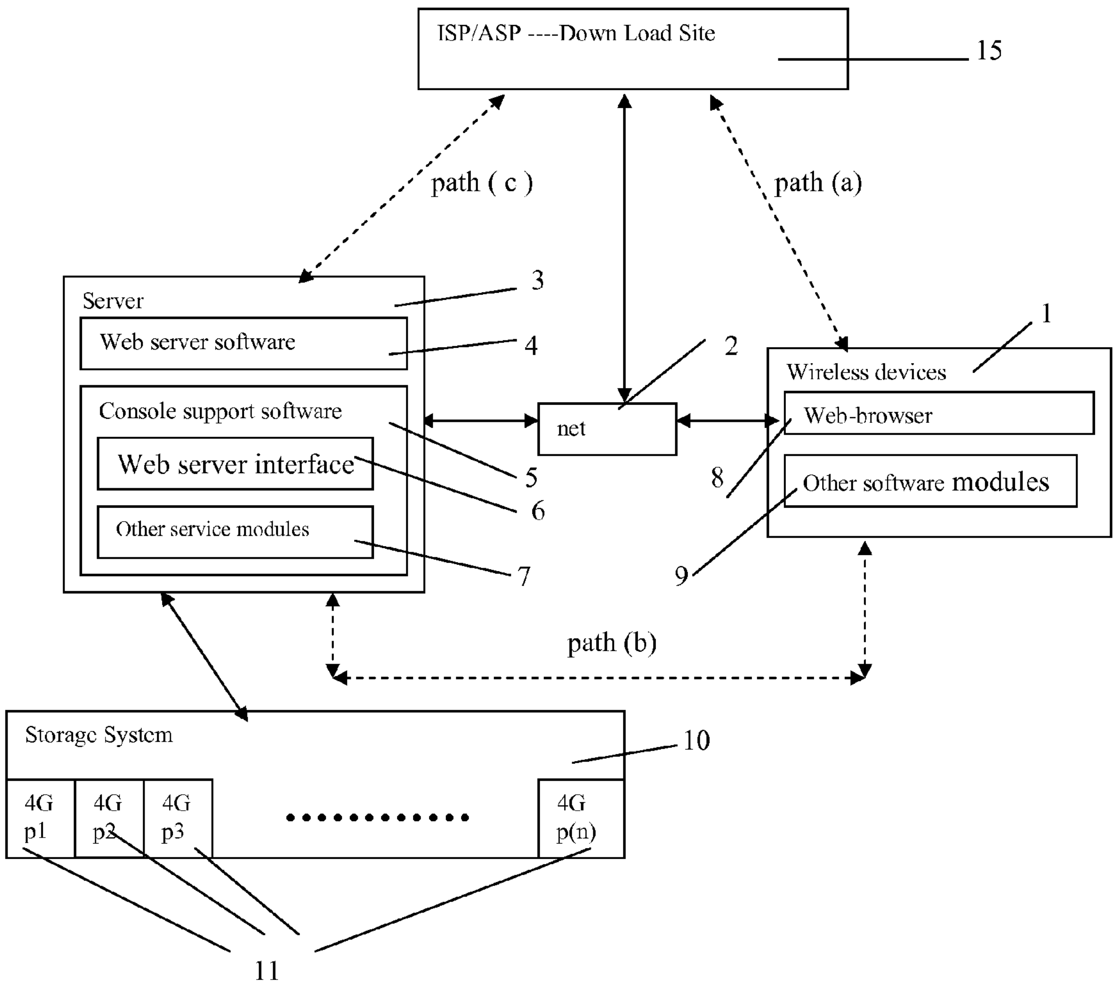


Fig. 3

The CCDSVM Support External Device for Huge Number of Wireless Device

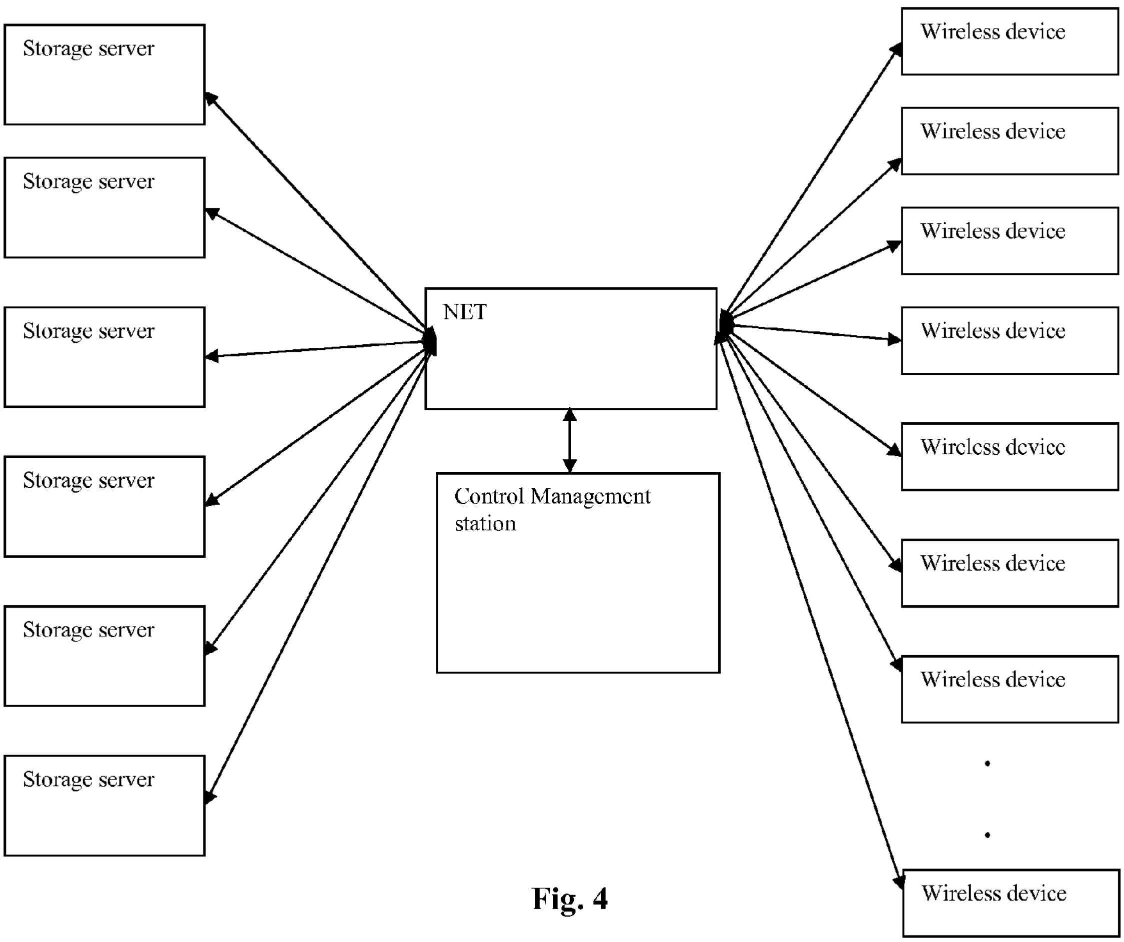


Fig. 4

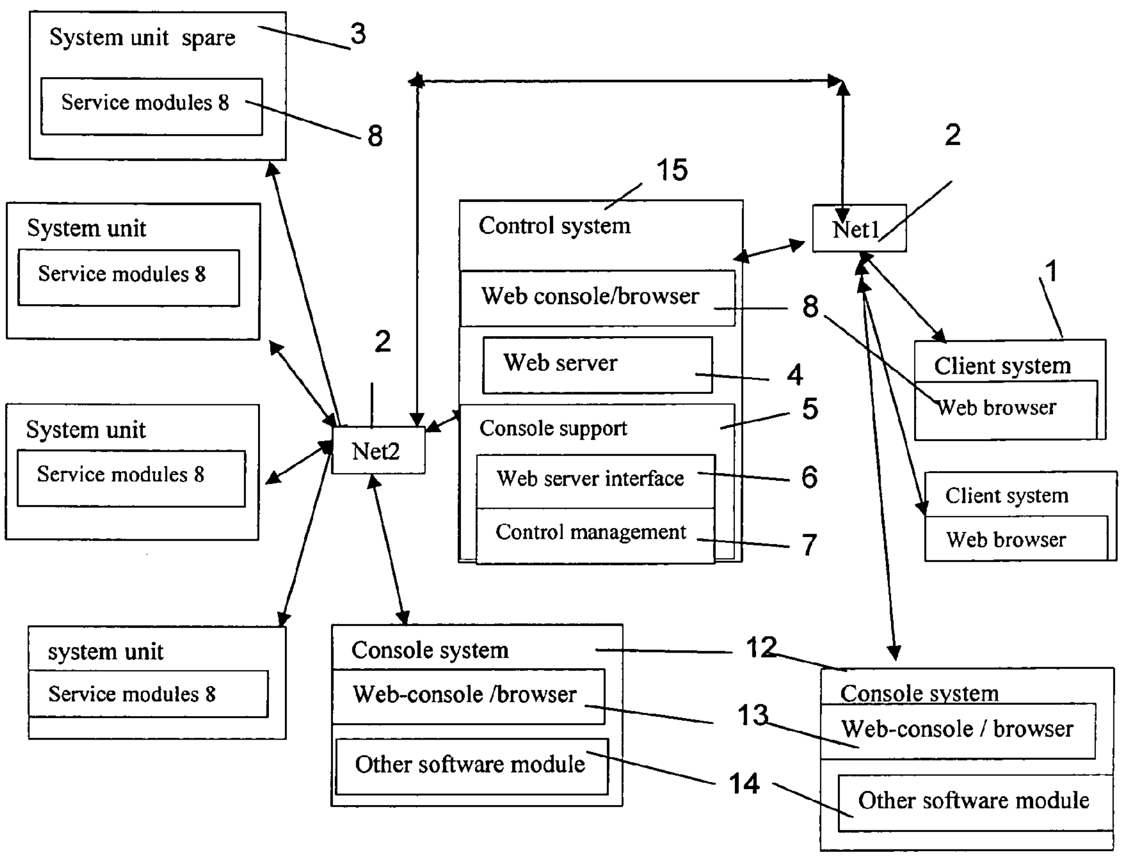

System unit spare
Service modules 8
3
8
System unit
Service modules 8
System unit
Service modules 8
system unit
Service modules 8
2
Net2
15
Control system
Web console/browser
Web server
Console support
Web server interface
Control management
8
4
5
6
7
Net1
2
1
Client system
Web browser
Client system
Web browser
Console system
Web-console /browser
Other software module
12
13
14
Console system
Web-console / browser
Other software module

Fig. 5

A typical Computer system connected into network

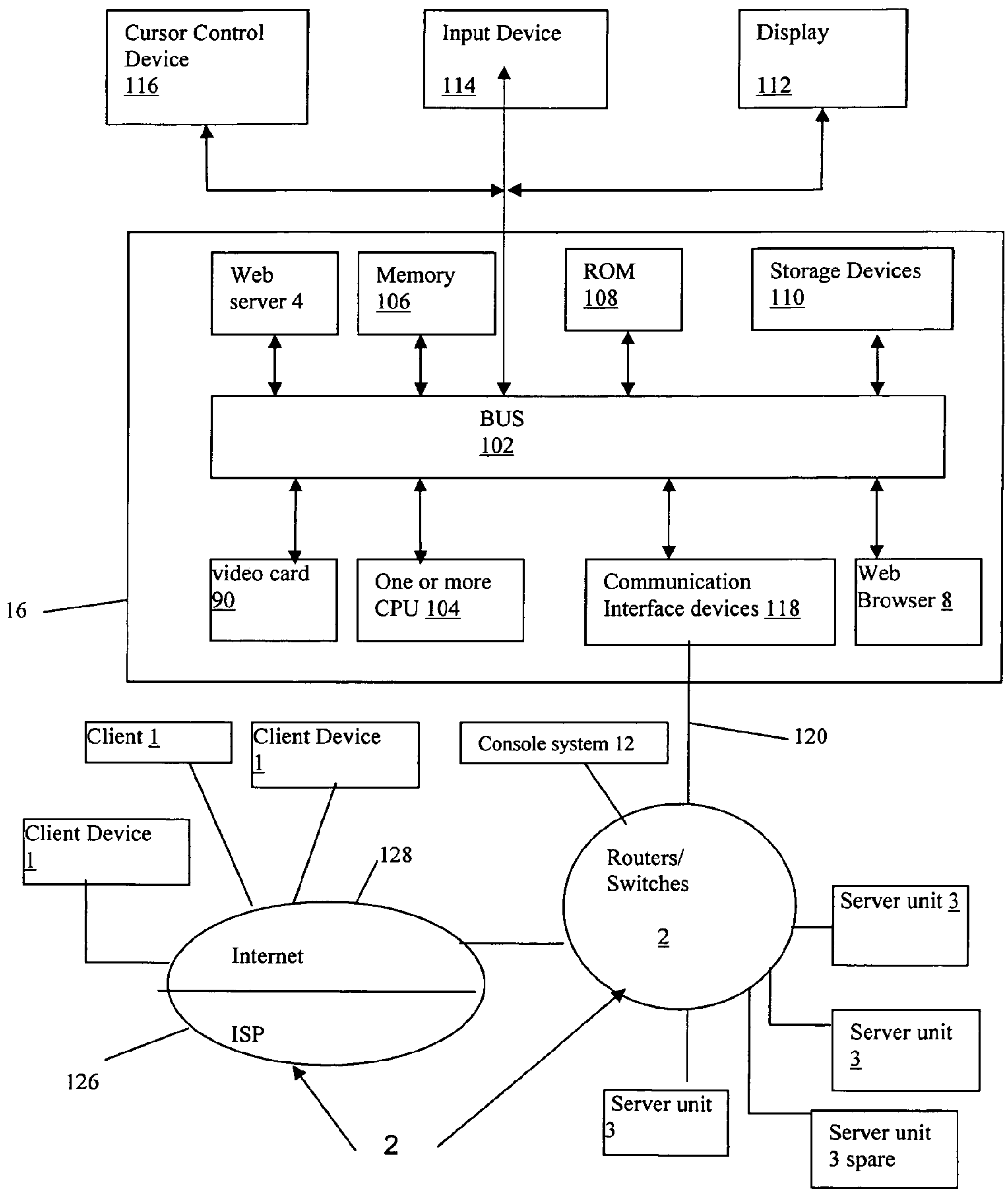


Fig. 6

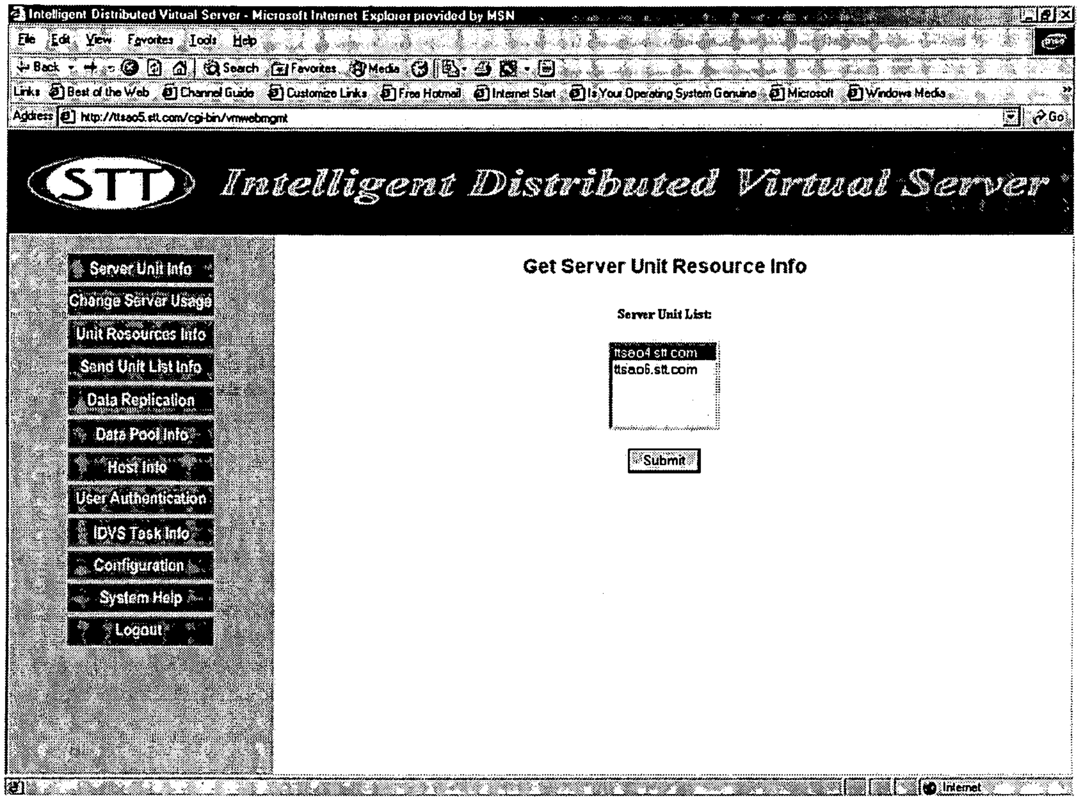

Intelligent Distributed Virtual Server - Microsoft Internet Explorer provided by MSN
http://ttsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Get Server Unit Resource Info
Server Unit List:
ttsao4.stt.com
ttsao6.stt.com
Submit
Internet

Fig. 7A

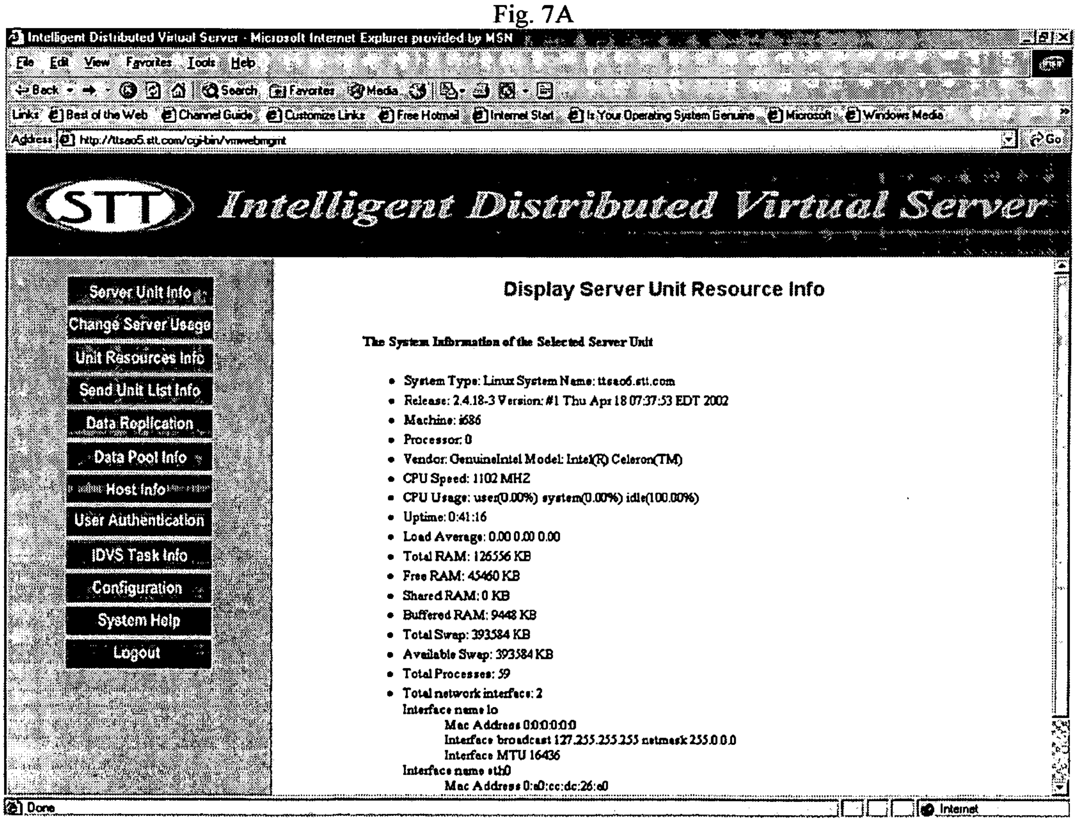

Intelligent Distributed Virtual Server - Microsoft Internet Explorer provided by MSN
http://ttsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Display Server Unit Resource Info
The System Information of the Selected Server Unit
System Type: Linux System Name: ttsao6.stt.com
Release: 2.4.18-3 Version: #1 Thu Apr 18 07:37:53 EDT 2002
Machine: i686
Processor: 0
Vendor: GenuineIntel Model: Intel(R) Celeron(TM)
CPU Speed: 1102 MHZ
CPU Usage: user(0.00%) system(0.00%) idle(100.00%)
Uptime: 0:41:16
Load Average: 0.00 0.00 0.00
Total RAM: 126556 KB
Free RAM: 45460 KB
Shared RAM: 0 KB
Buffered RAM: 9448 KB
Total Swap: 393584 KB
Available Swap: 393584 KB
Total Processes: 59
Total network interface: 2
Interface name lo
Mac Address 0:0:0:0:0:0
Interface broadcast 127.255.255.255 netmask 255.0.0.0
Interface MTU 16436
Interface name eth0
Mac Address 0:a0:cc:dc:26:e0
Done
Internet

Fig. 7B

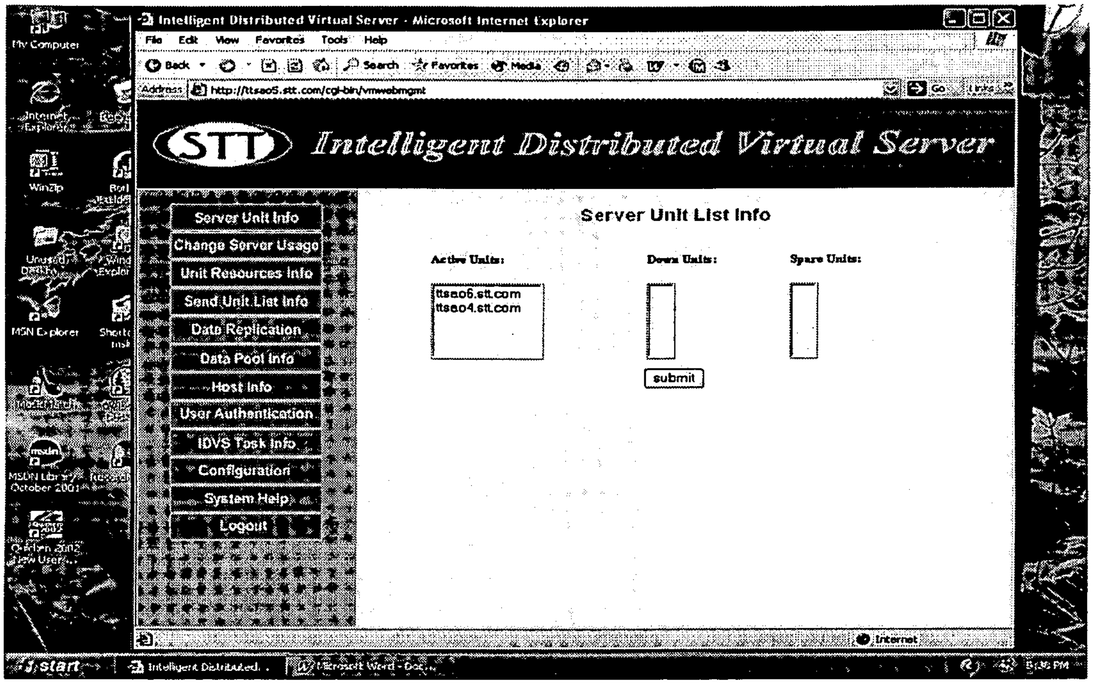

Intelligent Distributed Virtual Server - Microsoft Internet Explorer
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Server Unit List Info
Active Units:
Down Units:
Spare Units:
ttsao6.stt.com
ttsao4.stt.com
submit

Fig. 7C

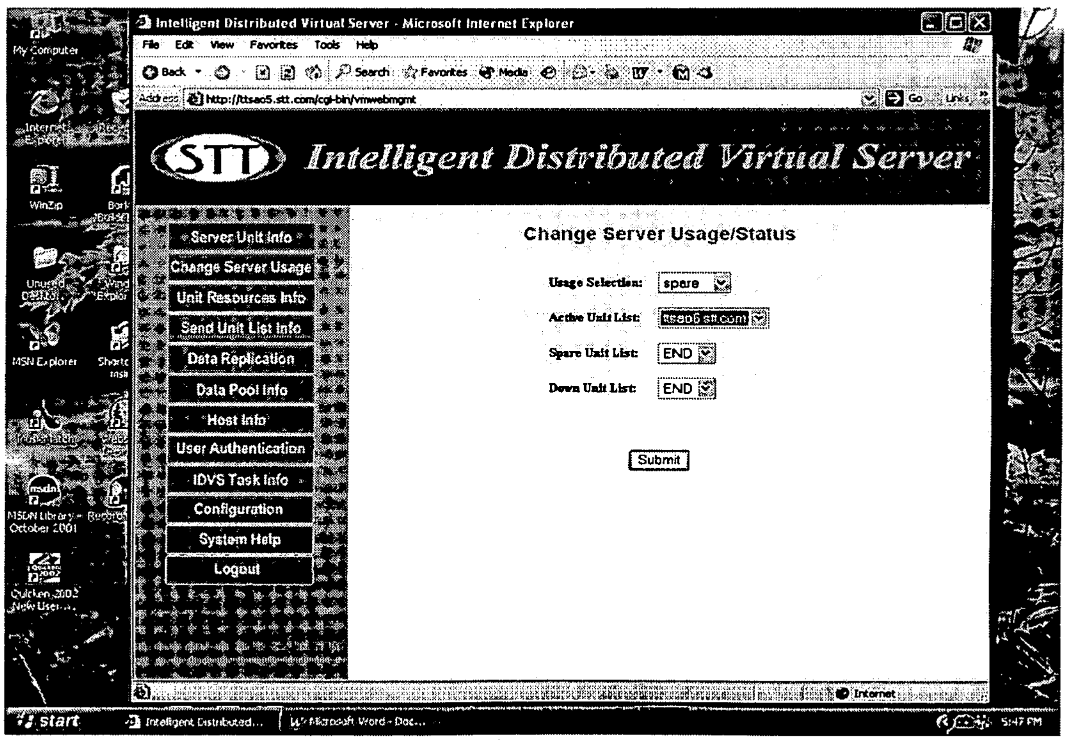

Intelligent Distributed Virtual Server - Microsoft Internet Explorer
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Change Server Usage/Status
Usage Selection:
spare
Active Unit List:
Spare Unit List:
END
Down Unit List:
END
Submit

Fig. 7D

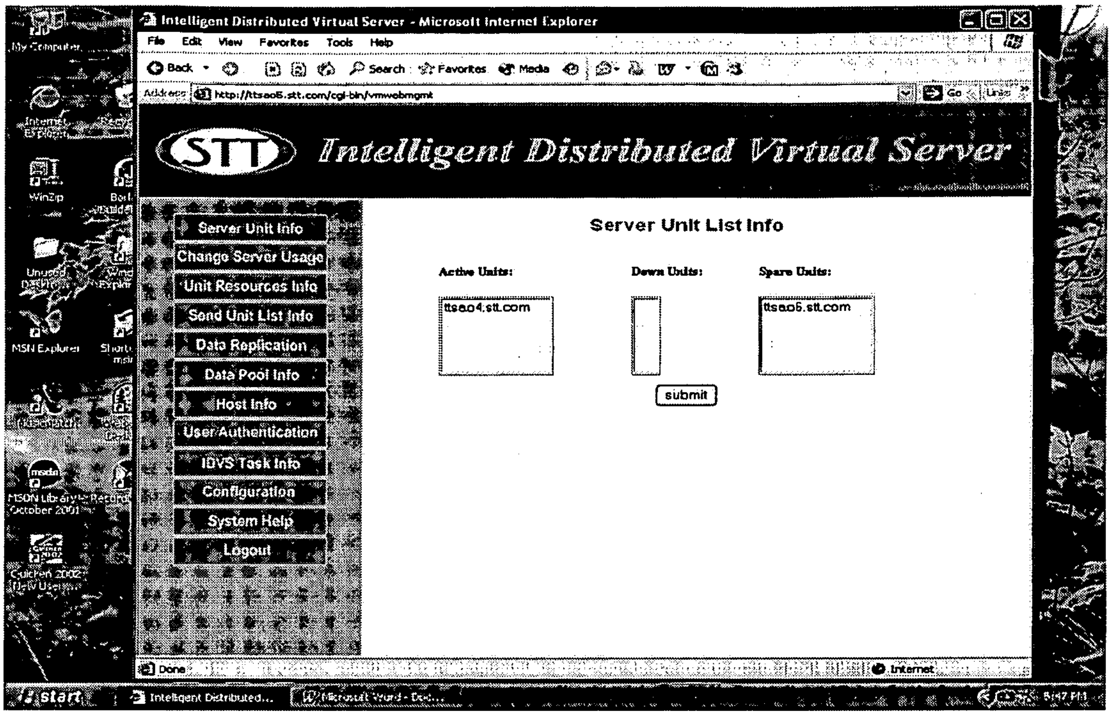

Intelligent Distributed Virtual Server - Microsoft Internet Explorer
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Server Unit List Info
Active Units:
Down Units:
Spare Units:
ttsao4.stt.com
ttsao6.stt.com
submit

Fig. 7E

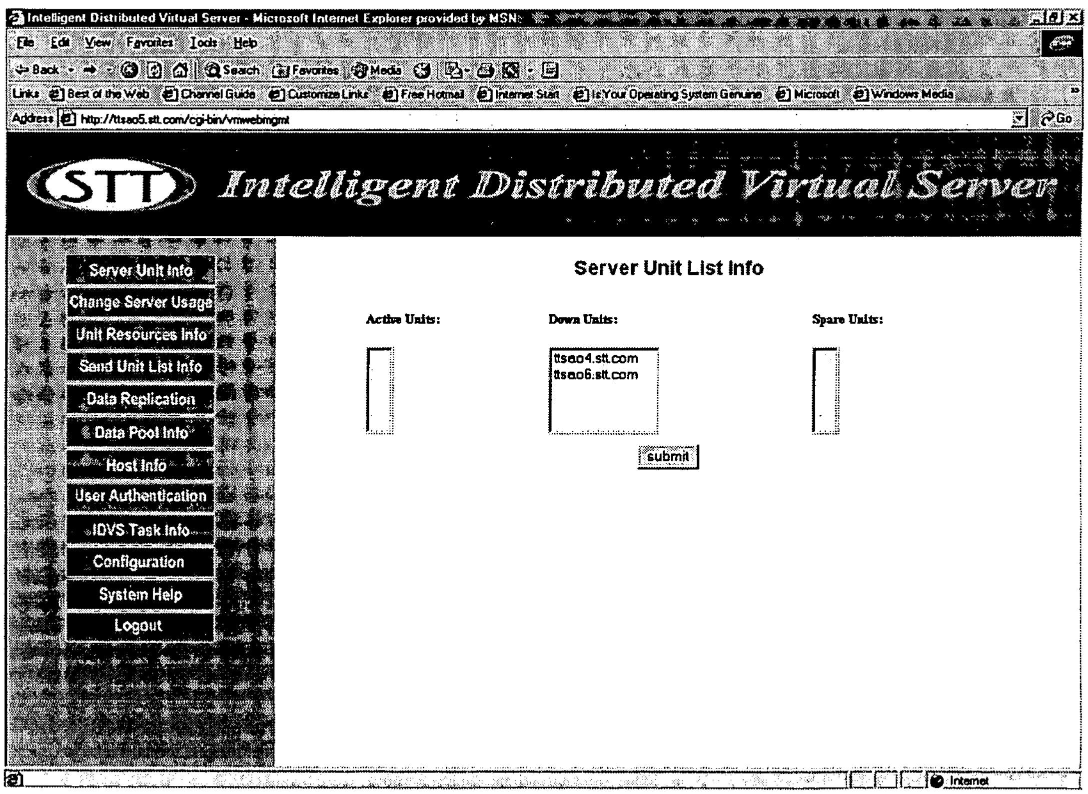

Intelligent Distributed Virtual Server - Microsoft Internet Explorer provided by MSN
http://ttsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Server Unit List Info
Active Units:
Down Units:
Spare Units:
ttsao4.stt.com
ttsao6.stt.com
submit

Fig. 7F

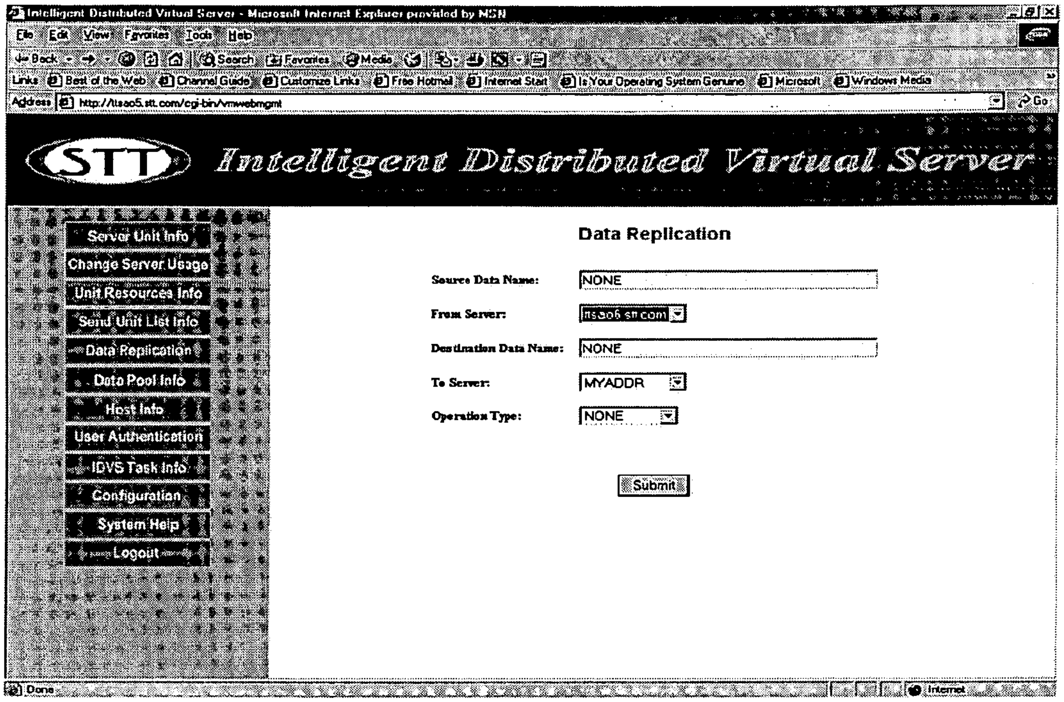

Intelligent Distributed Virtual Server - Microsoft Internet Explorer provided by MSN
http://ltsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Data Replication
Source Data Name:
NONE
From Server:
To Server:
MYADDR
Destination Data Name:
NONE
Operation Type:
NONE
Submit
Done
Internet

Fig. 8A

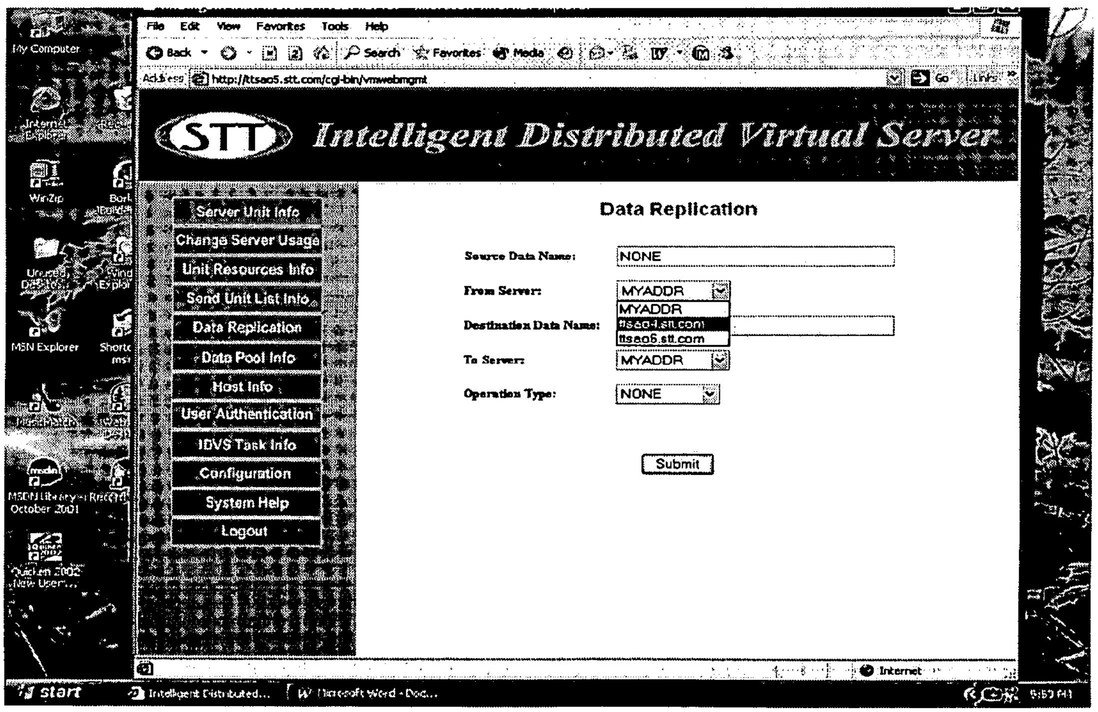

http://ltsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Data Replication
Source Data Name:
NONE
From Server:
MYADDR
MYADDR
ltsao6.stt.com
Destination Data Name:
To Server:
MYADDR
Operation Type:
NONE
Submit
start

Fig. 8B

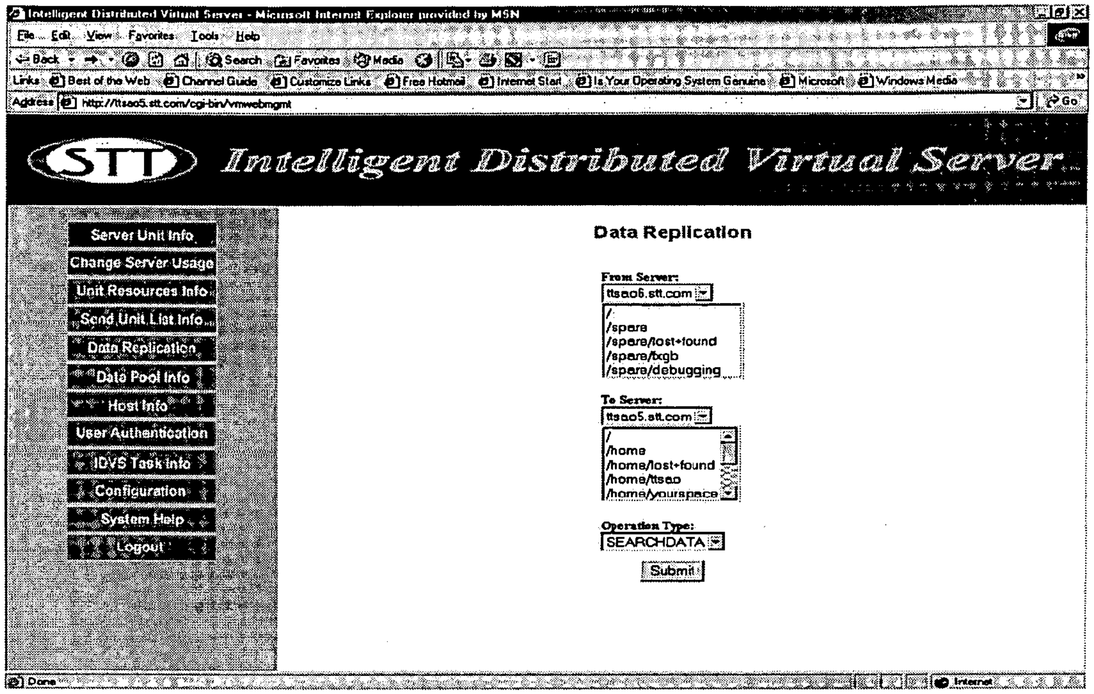

Intelligent Distributed Virtual Server - Microsoft Internet Explorer provided by MSN
http://ttsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Data Replication
From Server:
ttsao6.stt.com
/
/spare
/spare/lost+found
/spare/txgb
/spare/debugging
To Server:
ttsao5.stt.com
/
/home
/home/lost+found
/home/ttsao
/home/yourspace
Operation Type:
SEARCHDATA
Submit
Done
Internet

Fig. 8C

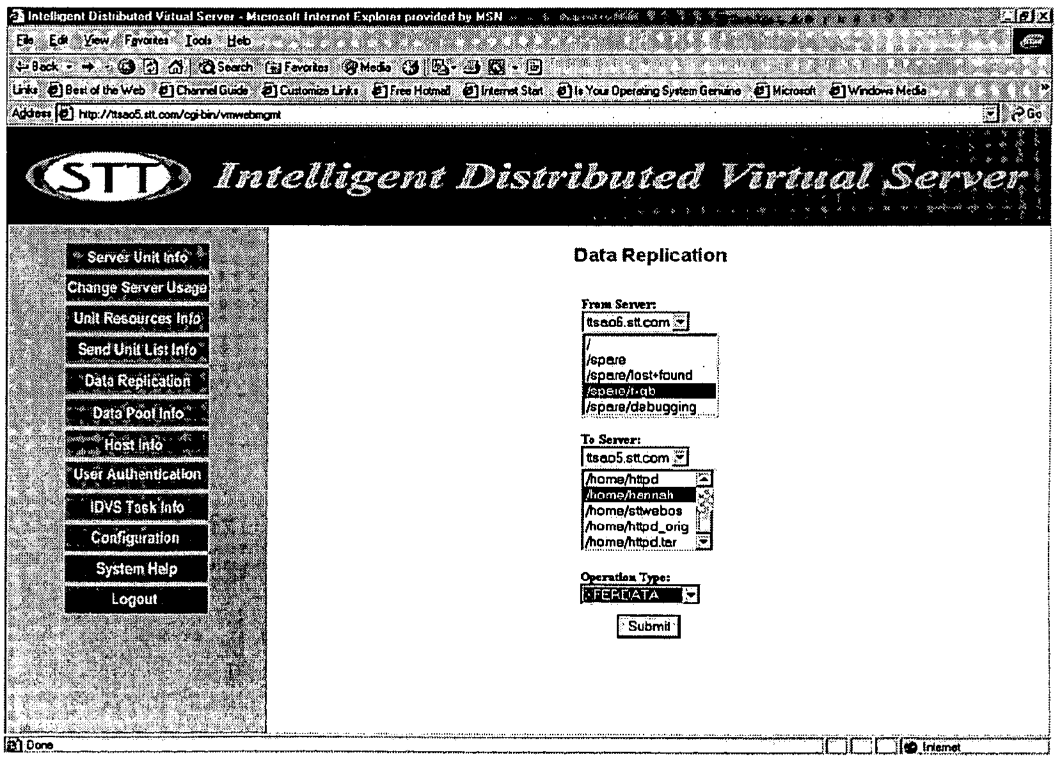

Intelligent Distributed Virtual Server - Microsoft Internet Explorer provided by MSN
http://ttsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Data Replication
From Server:
ttsao6.stt.com
/
/spare
/spare/lost+found
/spare/debugging
To Server:
ttsao5.stt.com
/home/httpd
/home/stwebos
/home/httpd_orig
/home/httpd.tar
Operation Type:
Submit
Done
Internet

Fig. 8D

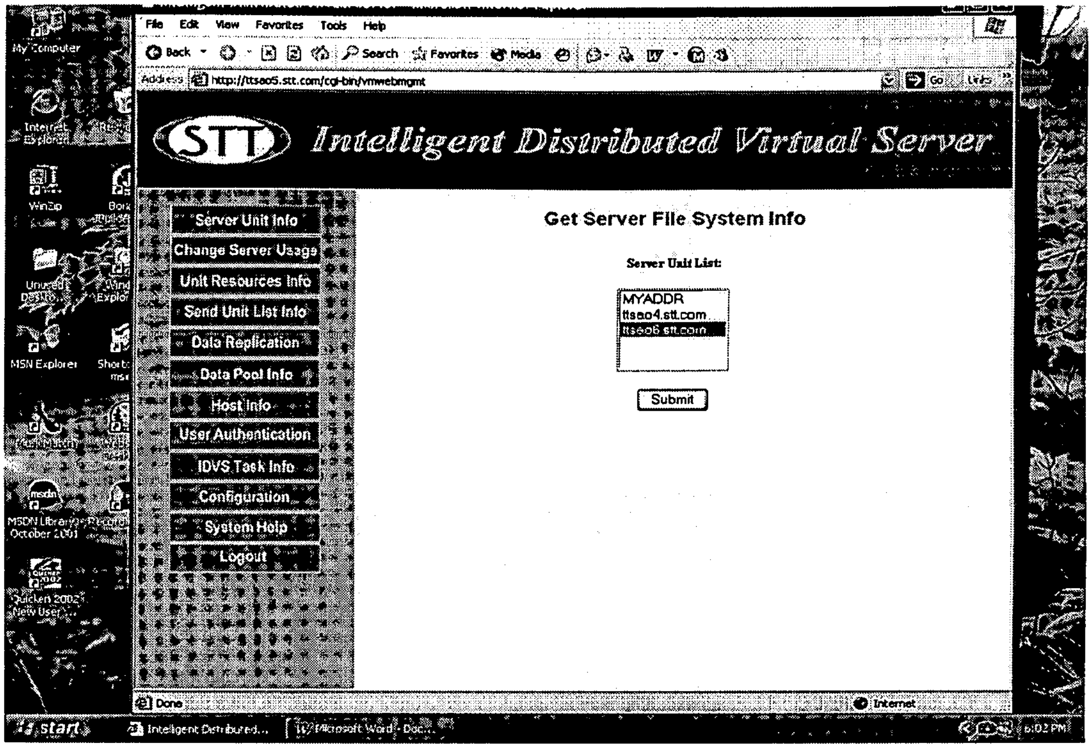

File Edit View Favorites Tools Help
http://ttsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Get Server File System Info
Server Unit List:
MYADDR
ttsao4.stt.com
ttsao6.stt.com
Submit
Done
Internet
start
Intelligent Distributed...
Microsoft Word - Doc...

Fig. 9A

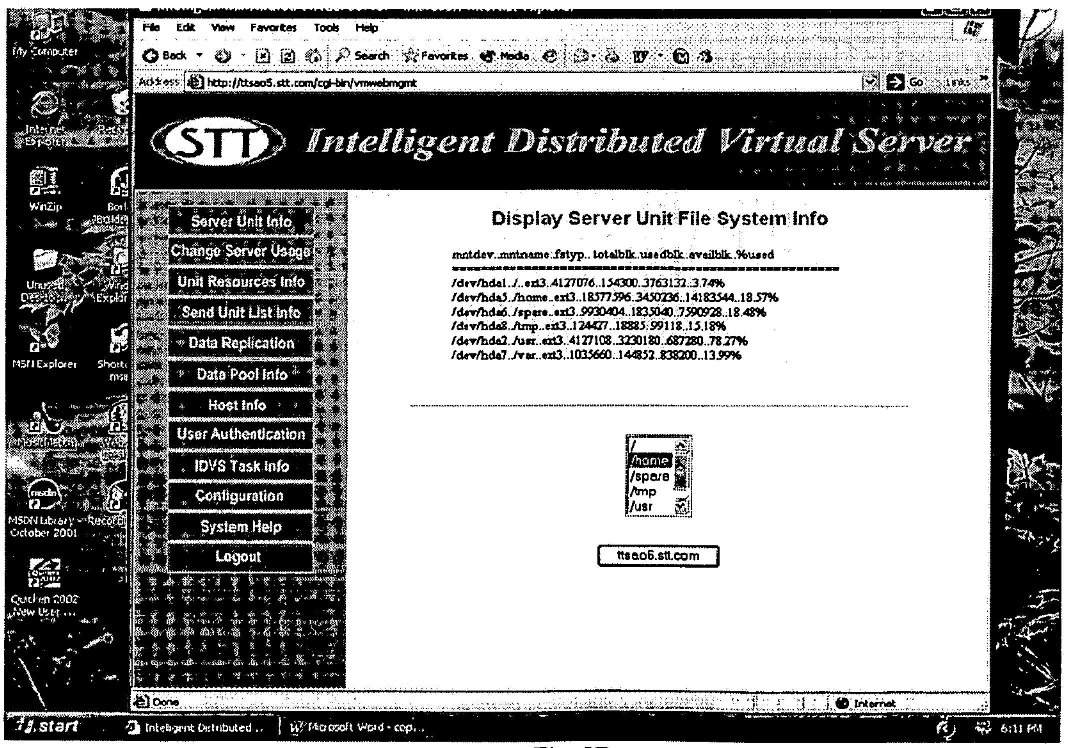

File Edit View Favorites Tools Help
http://ttsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Display Server Unit File System Info
mntdev..mntname..fstyp.. totalblk..usedblk..availblk..%used
/dev/hda1../..ext3..4127076..154300..3763132..3.74%
/dev/hda5../home..ext3..18577596..3450236..14183544..18.57%
/dev/hda6../spare..ext3..9930404..1835040..7590928..18.48%
/dev/hda8../tmp..ext3..124427..18885..99118..15.18%
/dev/hda2../usr..ext3..4127108..3230180..687280..78.27%
/dev/hda7../var..ext3..1035660..144852..838200..13.99%
/
/home
/spare
/tmp
/usr
ttsao6.stt.com
Done
Internet
start
Intelligent Distributed...
Microsoft Word - Doc...
6:11 PM

Fig. 9B

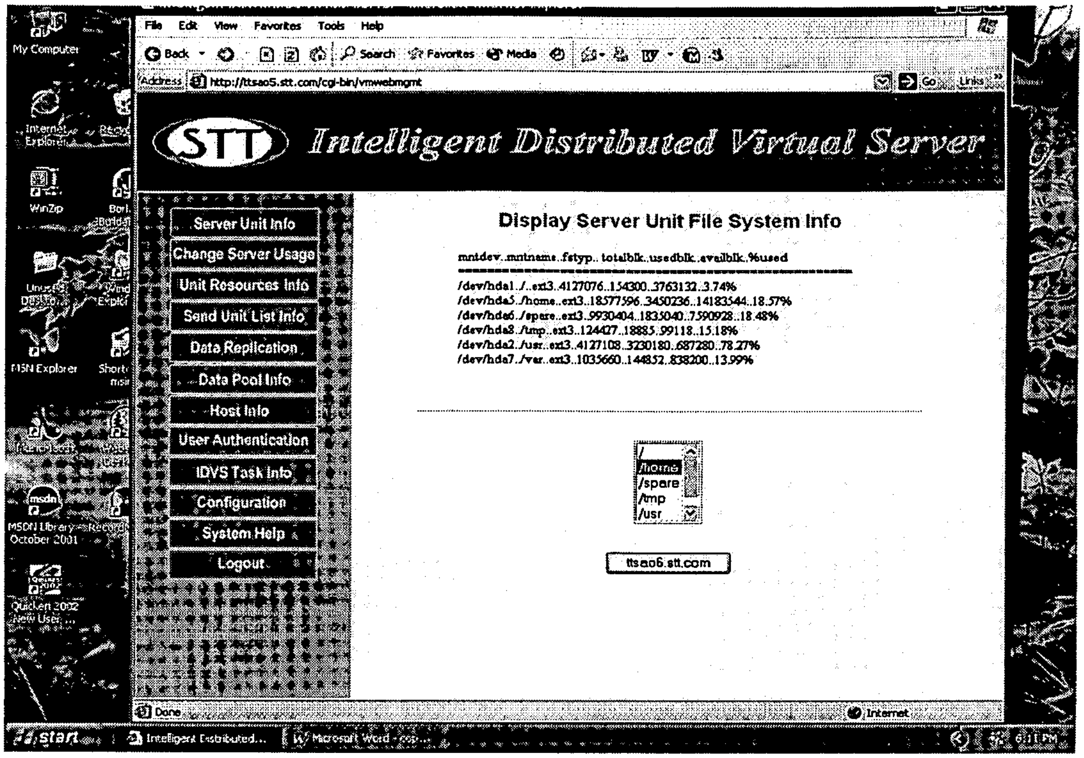

http://ttsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Display Server Unit File System Info
mntdev..mntname..fstyp.. totalblk..usedblk..availblk..%used
/dev/hda1../..ext3..4127076..154300..3763132..3.74%
/dev/hda5../home..ext3..18577596..3450236..14183544..18.57%
/dev/hda6../spare..ext3..9930404..1835040..7590928..18.48%
/dev/hda8../tmp..ext3..124427..18885..99118..15.18%
/dev/hda2../usr..ext3..4127108..3230180..687280..78.27%
/dev/hda7../var..ext3..1035660..144852..838200..13.99%
/
/home
/spare
/tmp
/usr
ttsao6.stt.com

Fig. 9C

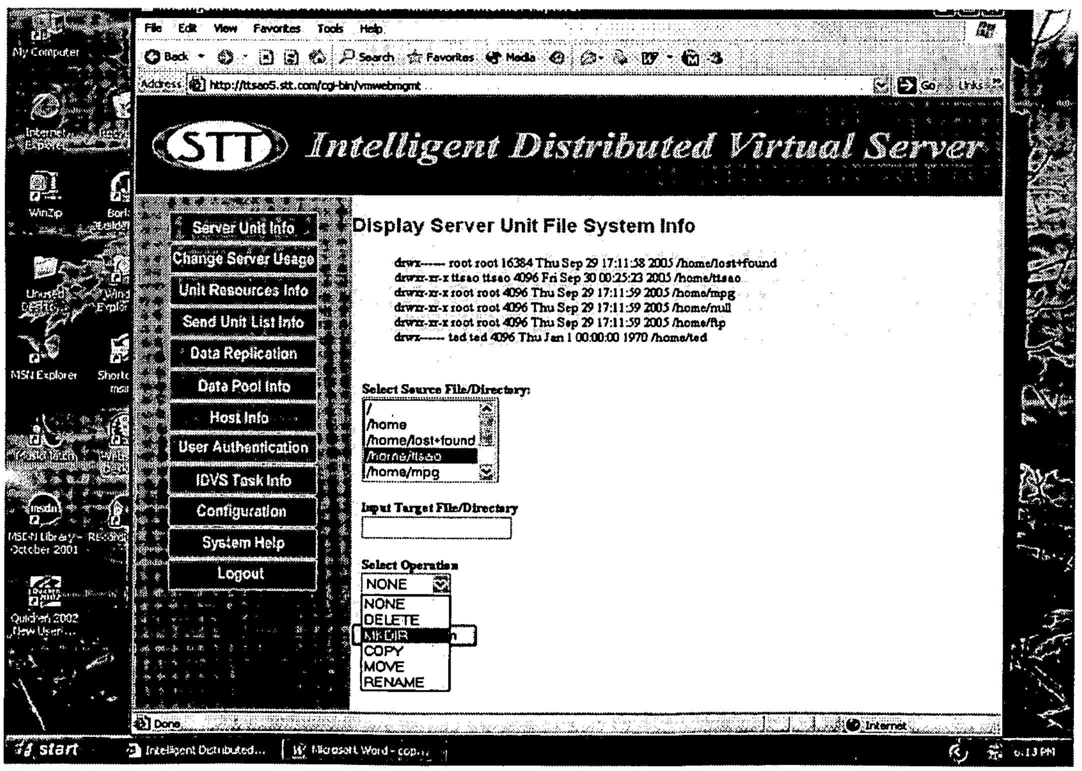

http://ttsao5.stt.com/cgi-bin/vmwebmgmt
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Display Server Unit File System Info
drwx------ root root 16384 Thu Sep 29 17:11:58 2005 /home/lost+found
drwxr-xr-x ttsao ttsao 4096 Fri Sep 30 00:25:23 2005 /home/ttsao
drwxr-xr-x root root 4096 Thu Sep 29 17:11:59 2005 /home/mpg
drwxr-xr-x root root 4096 Thu Sep 29 17:11:59 2005 /home/null
drwxr-xr-x root root 4096 Thu Sep 29 17:11:59 2005 /home/ftp
drwx------ ted ted 4096 Thu Jan 1 00:00:00 1970 /home/ted
Select Source File/Directory:
/
/home
/home/lost+found
/home/ttsao
/home/mpg
Input Target File/Directory
Select Operation
NONE
NONE
DELETE
MKDIR
COPY
MOVE
RENAME

Fig. 9D

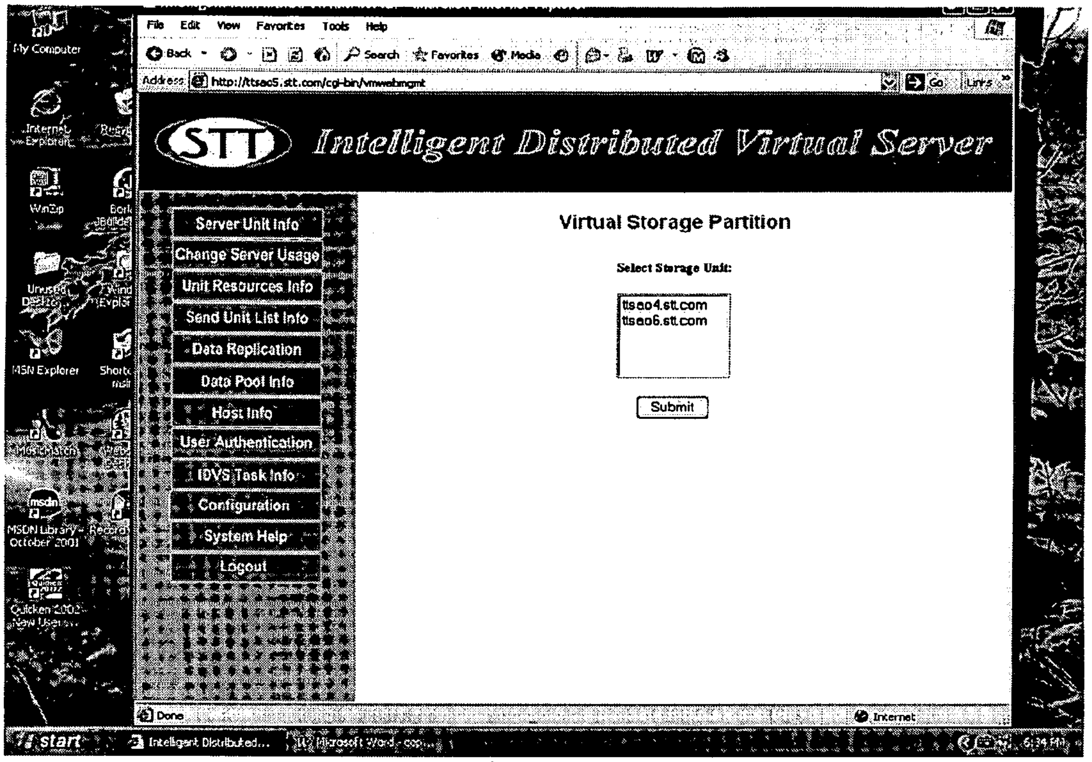


Fig. 10A

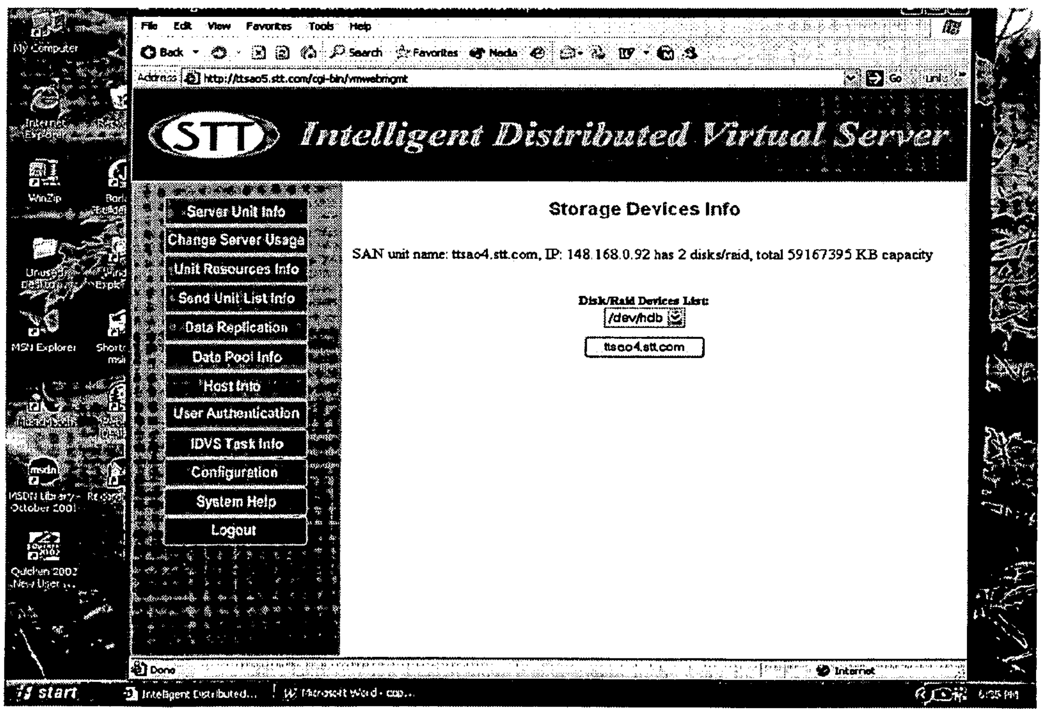


Fig. 10B

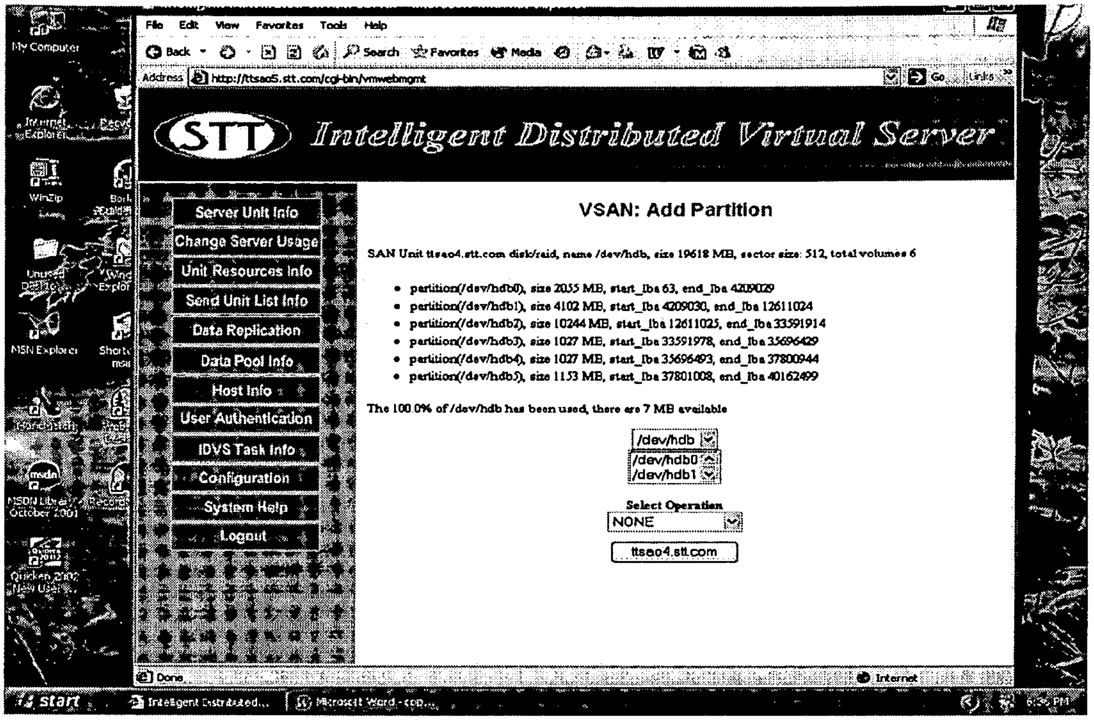


Fig. 10C

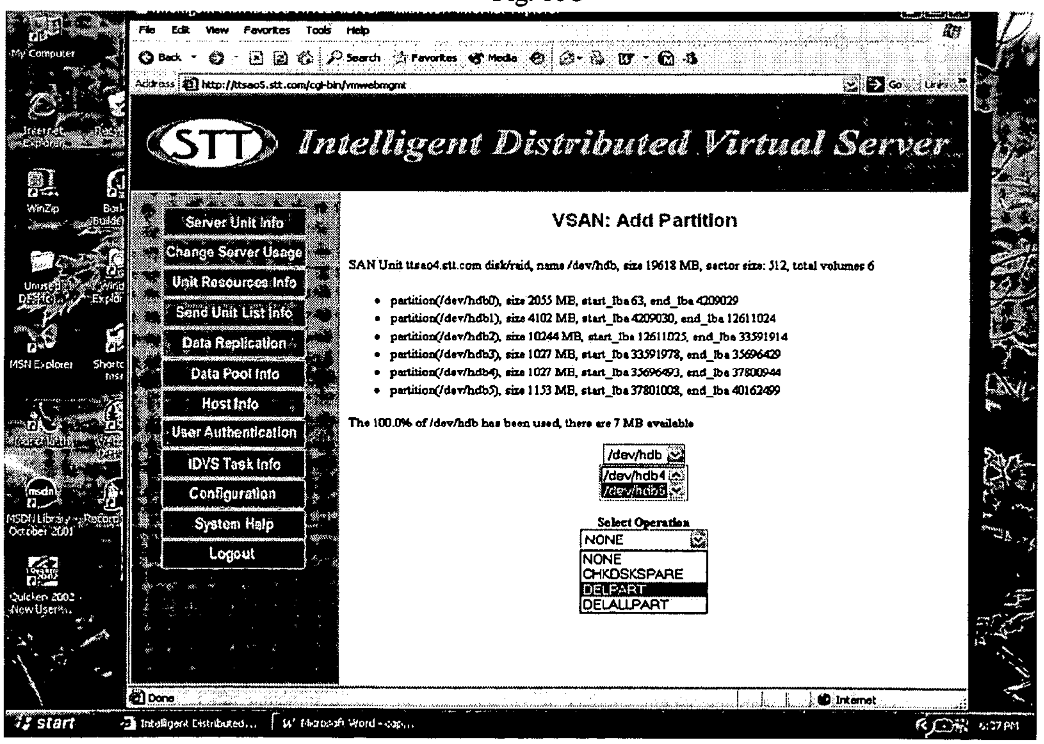


Fig. 10D

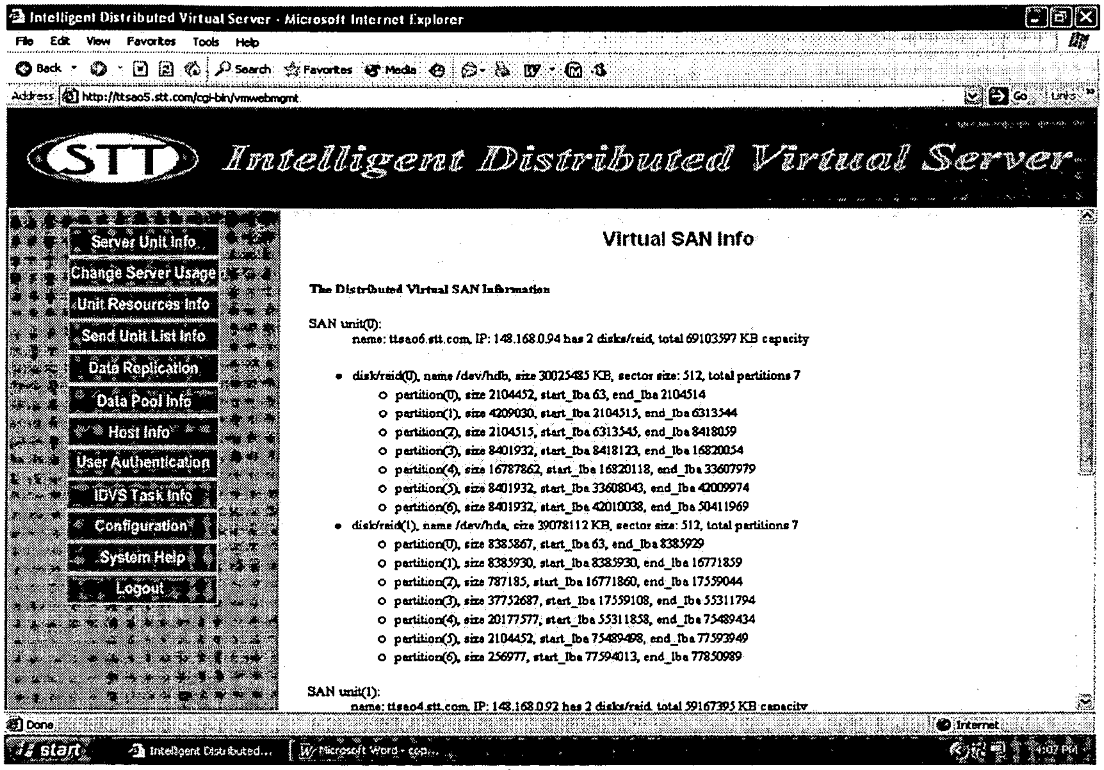


Fig. 10E

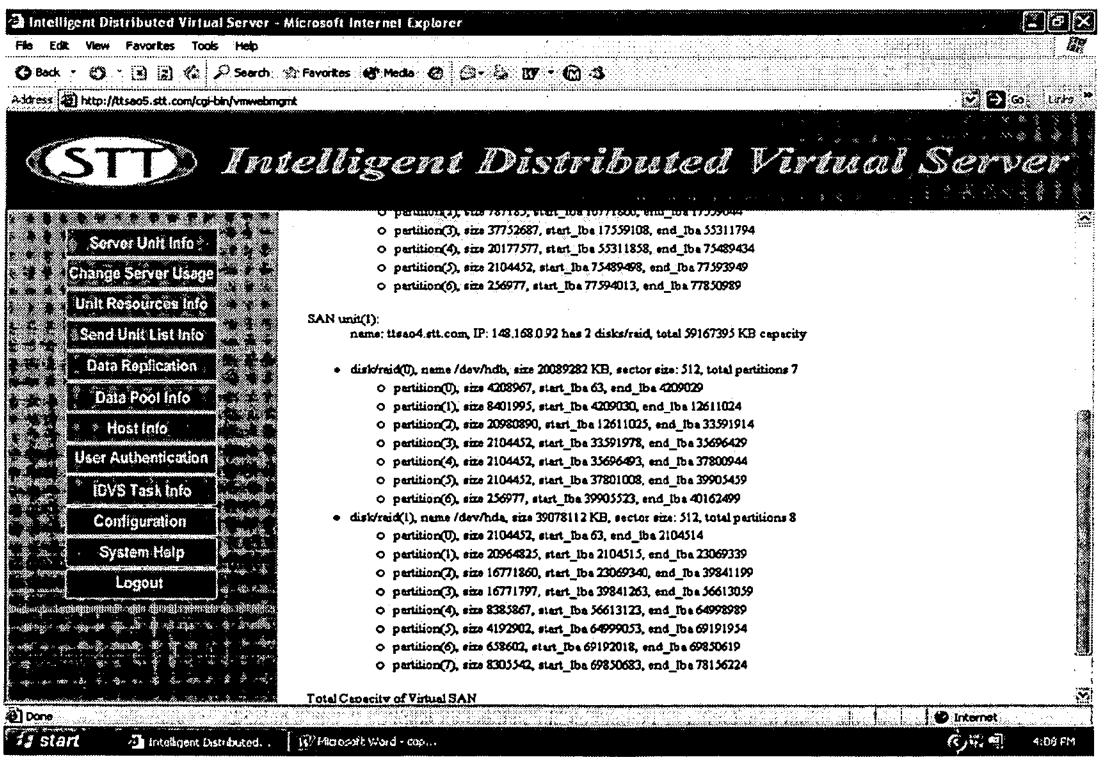


Fig. 10F

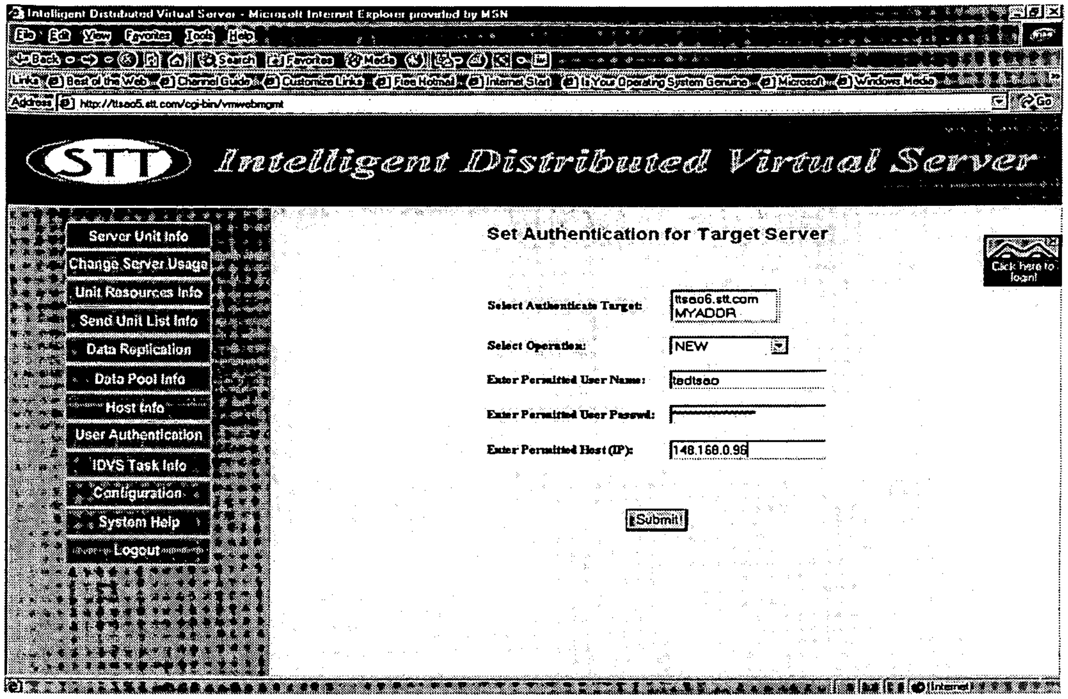


Fig. 11A

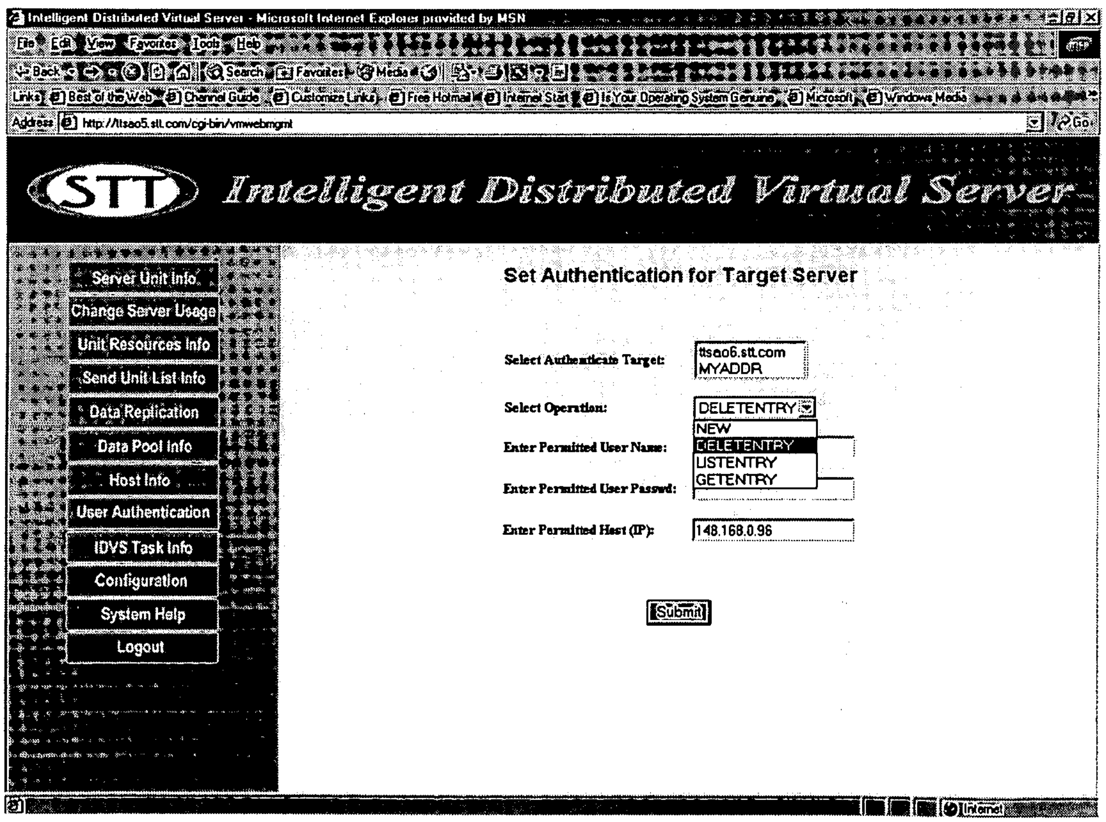


Fig. 11B

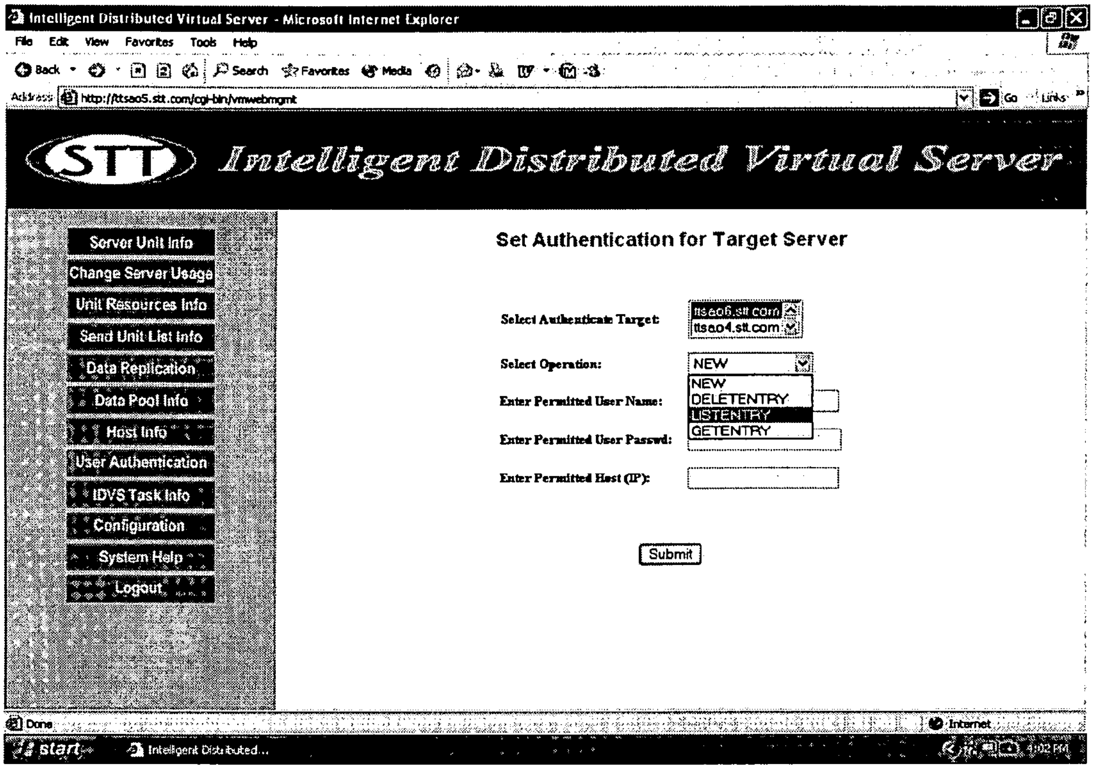

Intelligent Distributed Virtual Server - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://ttsao5.stt.com/cgi-bin/vmwebmgmt
Go Links
STT
Intelligent Distributed Virtual Server
Server Unit Info
Change Server Usage
Unit Resources Info
Send Unit List Info
Data Replication
Data Pool Info
Host Info
User Authentication
IDVS Task Info
Configuration
System Help
Logout
Set Authentication for Target Server
Select Authenticate Target:
ttsao6.stt.com
ttsao4.stt.com
Select Operation:
NEW
NEW
DELETENTRY
LISTENTRY
GETENTRY
Enter Permitted User Name:
Enter Permitted User Passwd:
Enter Permitted Host (IP):
Submit
Done
Internet
start
Intelligent Distributed...

Fig. 11C

USE OF WIRELESS DEVICES' EXTERNAL STORAGE

CROSS REFERENCE TO PRIOR APPLICATION

This invention relates to the previous invention, application No. 60/401,238 of "Concurrent Web Based Multi-task Support for Control Management System". This invention also relates to previous invention, application No. 60/402,626 of "IP Based Distributed Virtual SAN".

FIELD OF THE INVENTION

This invention focuses on how a wireless device can actually use external storage provided by a storage server. This invention also includes how a wireless device can download data to its external storage.

BACKGROUND INFORMATION

Terminology:

The terminologies described in next few sections reflect the scope and are part of present invention.

The Internal Storage of a System:

The storage media such as hard disk drives, memory sticks, memory etc. is connected to a system directly through bus or a few inches of cable. Therefore, the storage media actually is a component of the system in an enclosure.

The External Storage of a System:

The external storage media is not a component of the system in the same enclosure. Therefore, they have to be connected through a connecting medium (e.g. a cable) such as Ethernet cable for IP based storage, Fiber channel cable for fiber channel storage, or such as wireless medium and etc. The storage media of an external storage could be magnetic hard disk drives, solid state disk, optical storage drives, memory card, etc. and could be in any form such as Raid which usually consists of a group of hard disk drives.

The Storage Partition, its Volumes, and the Corresponding File System:

To effectively use storage system, each storage device usually needs to be partitioned into small volumes. After the partition, each of the volumes can be used to establish a file system on it. To simplify the discussion herein, the term of the storage volume, its corresponding file system, and the term of the partition of the storage device are often used without differentiation.

CCDSVM:

It is an abbreviation for a central controlled distributed scalable virtual machine system. The CCDSVM allows a control management station to control a group of systems and provide distributed services to a client system on the Internet, the Intranet, and an LAN environment.

ISP & ASP:

The ISP refers to Internet service provider and the ASP refers to application service provider.

FIGURES

FIG. **1** illustrates an embodiment of the instant application, the FIG. **1** is the same as FIG. **1** of the previous application of the "Concurrent Web Based Multi-task Support for Control Management System" with an exception of replacing a console host with a wireless device.

FIG. **2** is the same as FIG. **1** of the above except that it shows a more detailed storage system controlled by a server. In addition, multiple wireless devices are presented to access the storage system.

FIG. **3** shows a scheme of a wireless device downloading contents from an ISP/ASP or other web sites to an external storage allocated for the wireless device.

FIG. **4** is similar to the FIG. **1** of the previous application of the "IP Based Distributed Virtual SAN" with exception that each IP storage server provides a file system as external storage for each of the wireless devices instead of providing IP based virtual SAN service. Also, each host of mentioned FIG. **1** actually is replaced by a wireless device of present application.

Unless specified, the programming languages and the protocols used by each software modules of instant application, and the computing systems used in this invention are assumed to be the same as described in the previous patent applications.

In addition, in the drawing, like elements are designated by like reference numbers. Further, when a list of identical elements is present, only one element will be given the reference number.

BRIEF DESCRIPTION OF THE INVENTION

Today the wireless users commonly face a problem of lack of storage capacity on their wireless devices such as cell phone or PDA, which are usually limited to 256 MB for PDA and much less for cell phone. To effectively solve this problem and let users own multiple gigabytes (GB) of storage for their wireless devices as well as allowing the users to use the GB storage for their multimedia applications, the storage of a server can be used as the external storage for the wireless devices. This technology has been briefly introduced in the previous parent patent applications.

Now let us examine how the external storage can actually be used by the wireless devices. First, let each server unit (e.g. the server **3** of the FIG. **2**) partitions its storage system into volume and each of the volumes will have multiple GB in size. Therefore, each user of the wireless devices can be exclusively assigned and access a specific storage volume. For example, if we need to provide each user a 4 GB storage space, then a 160 GB disk drive can support 40 users. Therefore, a 4096 GB storage system on the server unit can support a total of 1024 wireless devices for users. Further, any data on the wireless device can be transmitted to an assigned storage volume. In addition, the user of the wireless device also can download the multimedia data from an ISP or ASP to the assigned storage volume of a designated server unit through out-band approach shown in FIG. **3**. Finally, the user can use a web-browser, which has a functionality of invoking embedded video or music, to enjoy his/her stored multimedia contents.

These and other futures, aspects and advantages of the present invention will become understood with reference to the following description, appended claims, and accompanying figures:

DESCRIPTION OF THE DRAWINGS

Referring now to FIG. **1**, it demonstrates a configuration comprising a network connecting a wireless device and a server. In the FIG. **1**, Net (**2**) represents a communication link, which may be combined with wireless and wired connection media and guarantee that the communication packets can be sent/received between the wireless device and the server. It is also assumed that the net (**2**) representing an communication infrastructure is built up in such way that a user of a wireless device can access and browse any web-site on the Internet, the Intranet, or a local area network (LAN).

In FIG. **1**, the console support software (**5**) on the server (**3**) can be configured to support web-based multi-tasks for the user of the wireless device (**1**) via a web browser **8**. Further, the user of the wireless device is able to perform creating structured layered files, directories, or folders, and perform data management operations, such as delete, move, copy, rename for data files or folders, directories and etc. on an assigned storage volume associated with the server (**3**).

In addition, the other software modules (**9**) of the wireless device (**1**) is configured to send data to or receive data from the other service modules (**7**) running on the server (**3**) via communication link (**2**) through a suitable IP or non-IP based protocol. The data being sent cold be a digital photo picture, a message and etc.

Also, the console supporting software (**5**) of the server (**3**) and the other software modules (**9**) of the wireless device (**1**) can be implemented with any suitable languages such as C, C++, Java, etc. depending on the implementation.

Besides, the web-browser (**8**) of the wireless device (**1**) can be implemented any suitable software. The web browser is configured to communication with web server software (**4**) on the server (**3**) with any other web server through the HTTP protocol.

On the other hand, FIG. **2** has demonstrated that the storage system **10** of a server **3** can be allocated to multiple wireless devices. First, the storage system (**10**) of the server (**3**) can be partitioned into multiple storage volumes (**11**) by administration staff through a web-console (**13**) of a console host (**12**).

Second, the storage system (**10**) of the server (**3**) can be partitioned in such way that each of the wireless devices can be allocated with a storage volume having a desired size, therefore, the server **3** can support maximum numbers of the wireless devices.

In addition, the storage connection media could be any kind of cables, such as SCSI cable, IP cable, Fiber cable etc. or could be wireless communication media. The storage system itself could be various types.

Finally, the storage system **10** can be accessed by each of the wireless devices through IP or non-IP based network and protocols.

FIG. **3** has demonstrated that a user from a web-browser (**8**) on a wireless device (**1**) can download data from a known web-site (**12**) to his/her allocated external storage (**10**) on the server (**3**). The dash-lined path (a) represents a communication channel between the wireless device (**1**) and a remote downloading web-site (**12**) that provides downloading contents. The dash-lined path (b) represents a communication channel between the wireless devices (**1**) and the storage server (**3**). The dash-lined path (c) represents a communication channel between the server **3** and the remote web-server (**12**).

THE DETAILED DESCRIPTION OF THE INVENTION

The Use of the External Storage by the Wireless Device:

The FIG. **2** shows a simplified diagram of the wireless devices (**1** of FIG. **2**) using the external storage system (**10** of FIG. **2**) of the server (**3** of FIG. **2**) for effectively resolving the storage limitation problem for the wireless devices (**1** of FIG. **2**).

Partition Storage Volumes (FIG. **2**)

With this invention, the entire storage (**10** of FIG. **2**) on the server (**3** of FIG. **2**) needs to be partitioned into suitable size of volumes (**11** of FIG. **2**) such as 4 GB for each volume. This will allow the server **3** to serve maximum number of the wireless devices (**1** of FIG. **2**). With the web console support software (**5** of FIG. **2**) of the server (**3** of FIG. **2**), tasks of partitioning the storage system **10** can be done through a web-console (**13** of FIG. **2**) on a console host (**12** of FIG. **2**) by an administrative staff.

In order to support storage partitioning, first the console support software (**5** of FIG. **2**) of the server (**3** of FIG. **2**) must send storage information of the server (**3** of FIG. **2**) to the web-console (**13** of FIG. **2**) of the console host (**12** of FIG. **2**). The storage information includes each storage device's name and total size etc. Second, based on the received storage information the administration staff on the console host (**12** of FIG. **2**) can use a web-console (**13** of FIG. **2**) to fill out and send the storage partition information to the console support software (**5** of FIG. **2**) of the server (**3** of FIG. **2**). The storage partition information includes the number of the partitions (volumes) and the size of each partition (volume). Third, upon receiving storage partition information from the web-console (**13** of FIG. **2**) of the console host (**12** of FIG. **2**), the console support software (**5** of FIG. **2**) of the server (**3** of FIG. **2**) performs the actual storage partition to divide the entire storage into multiple small volumes. Finally, for each small storage volume, a corresponding file system could be built up.

Assign Storage Volumes (FIG. **2**):

Each of the storage volumes together with its corresponding file system (**11** of FIG. **2**) on the storage system (**10** of FIG. **2**) of the server (**3** of FIG. **2**) needs to be exclusively allocated and exported to a specific wireless device (**1** of FIG. **2**) by the console support software (**5** of FIG. **2**) of the server (**3** of FIG. **2**).

Data and Storage Volume Management (FIG. **2**)

With the support of the console support software modules (**5** of FIG. **2**) of the server (**3** of FIG. **2**), the user of the wireless device (**1** of FIG. **2**) can via the web-browser **8** of FIG. **2** setup folder/directory structure on the file system of his/her assigned external storage volume (**11** of FIG. **2**). In addition, the user of the wireless device (**1** of FIG. **2**) can via the web-browser **8** of FIG. **2** perform all data management operations such as delete, copy, move, rename etc. for file system.

In order to support such data management on the external storage (**10** of FIG. **2**) allocated to the wireless device (**1** of FIG. **2**) via the web-browser **8** of FIG. **2**, first the console support software modules (**5** of FIG. **2**) of the server **3** of FIG. **2** must communicate with the web-browser (**8** of FIG. **2**) of the wireless device (**1** of FIG. **2**). Therefore, the user from the web-browser (**8** of FIG. **2**) of the wireless device (**1** of FIG. **2**) can choose desired data management operations and send operation information to the console support software modules (**5** of FIG. **2**) of the server **3** of FIG. **2**. The mentioned operations include establishing folder/directory, copying, moving, or reaming data file etc. Second, upon receiving the data management operation, the console support software modules (**5** of FIG. **2**) of the server **3** of FIG. **2** actually process/executes these requested operations for the assigned file system of an allocated storage volume (**11** of FIG. **2**) on the server **3** of FIG. **2**.

Store Data from Wireless Device into External Storage (FIG. **2**)

To store the data such as digital photo pictures or messages into the file system on the allocated storage (**10** of FIG. **2**) of the server (**3** of FIG. **2**), the other software modules (**9** of FIG. **2**) of the wireless device (**1** of FIG. **2**) need to send these data to the other service modules (**7** of FIG. **2**) of the server (**3** of FIG. **2**) via communication link between them. Upon receiving the data, the other service modules (**7** of FIG. **2**) of the server (**3** of FIG. **2**) write these data to the file system of the allocated storage volume (**11** of FIG. **2**) for the wireless device. The protocol used between these two communication entities could be either IP or non-IP based protocol.

Download Data from a Remote Web Server Site into Allocated Storage Volume (FIG. **3**)

If a user of the wireless device (**1** of FIG. **3**) wants to download a data from a remote web server (**12** of FIG. **3**) into the allocated file system (**11** of FIG. **3**) of the allocated storage (**10** of FIG. **3**) on the server (**3** of FIG. **3**), the following steps are required:

1) The user of the wireless device (**1** of FIG. **3**) via a web-browser (**8** of FIG. **3**) accesses a remote downloading web server site (**12** of FIG. **3**) to obtain information for the downloading via path (a) of FIG. **3**. For example, the user accesses a web-page which contains the data name for the downloading.

2) The other software modules (**9** of FIG. **3**) of the wireless device (**1** of FIG. **3**) obtain the downloading information, which becomes available in the cached web-pages on the wireless device (**1** of FIG. **3**) after the web-browser (**8** of FIG. **3**) accessing the web site (**12** of FIG. **3**).

3) The other software modules (**9** of FIG. **3**) of the wireless device (**1** of FIG. **3**) send the obtained downloading information to other service modules (**7** of FIG. **3**) of the storage server (**3** of FIG. **3**) via path (b) of FIG. **3**.

4) Upon receiving the downloading information from the wireless device (**1**), the other service module (**7** of FIG. **3**) of the storage server (**3** of FIG. **3**) sends a web download request to the web-site (**12** of FIG. **3**) via path (c) of FIG. **3** and receives the downloading data streams from the web server of the web-site (**12** of FIG. **3**).

5) Upon receiving downloading data streams, the other service modules (**7** of FIG. **3**) of the storage server (**3** of FIG. **3**) write the data streams into the allocated file system (**11** of FIG. **3**) on the server (**3** of FIG. **3**) for the wireless device (**1** of FIG. **3**).

Retrieve Data from Allocated Storage for the Wireless Device

1) If a web-browser (**8** of FIG. **2**) on a wireless device **1** of FIG. **2** has embedded video or music functionality, a user of the wireless device (**1** of FIG. **2**) can use the browser to retrieve and play multimedia data file such as video or music stored in the allocated storage volume (**10** of FIG. **2**) located on the server (**3** of FIG. **2**).

2) If there is a need, the other software module (**9** of FIG. **2**) of the wireless device (**1** of FIG. **2**) also can retrieve data file from the allocated file system of the allocated storage volume (**11** of FIG. **2**) on the server (**3** of FIG. **2**).

Support External Storage for a Large Number of the Wireless Devices

If there is a need to provide each user a 2 GB of storage space, then a 160 GB disk drive can support 80 users. A 4096 GB (4 Tera Bytes) storage system on the server unit can support 2024 user. Each of the server units only can efficiently support a limited size of the storage system. In order to support a large number of the wireless devices, such as for 500,000 wireless devices, a larger number of the servers is required, in this case 250 servers is required. In order to let a larger number of the servers to effectively support the larger number of the wireless devices, an infrastructure like CCDSVM is desirable, which has been described in previous patent applications. With the CCDSVM the control management station can control larger number of storage servers to provide external storage for a huge number of the wireless devices.

What is claimed is:

1. A method for expanding storage capacity of a wireless device, the method comprising:

allocating via a server a storage space of a predefined capacity for the wireless device, the storage space being remotely located with respect to the wireless device;

creating a file system for the storage space allocated for the wireless device;

establishing a link for the wireless device access to the storage space; and

updating the file system whenever a user of the wireless device performs an operation to the storage space,

wherein the updating of the file system comprises

updating the file system for storing a file therein, the storing of a file including to download a file from a remote web server, according to download information for the file cached in the wireless device received therefrom, directly into the storage device when the user via a web browser executed on the wireless device to perform an operation of downloading the file from the remote web server to the storage space instead of downloading the file into the wireless device itself.

2. A system for expanding storage capacity of a plurality of wireless devices, the system comprising:

a server configured to:

allocate a storage space of a predefined capacity for each of the wireless devices, create a file system for the storage space allocated for the each of the wireless devices;

establish a link for the each of the wireless devices access to the storage space; and

update the file system whenever a user of the each of the wireless devices performs an operation to the storage space, wherein the storage space being remotely located with respect to the each of the wireless devices,

wherein the updating of the file system comprises

updating the file system for storing a file therein, the storing of a file including to download a file from a remote web server, according to download information for the file cached in the each of the wireless devices received therefrom, directly into the storage device allocated thereto when the user via a web browser executed on the each of the wireless devices to perform an operation of downloading the file from the remote web server to the storage space instead of downloading the file into the each of the wireless devices itself; and

the wireless devices, wherein each of the wireless devices is operable access to the storage space allocated to the each of the wireless devices.

3. The method as recited in claim **1**, wherein the operation to the storage space comprises creating from the wireless device a folder in the storage space.

4. The method as recited in claim **1**, wherein the operation to the storage space comprises deleting or moving or copying or renaming, from the wireless device, a file or a folder being stored in the storage space.

5. The method as recited in claim **1**, wherein the link is wireless.

6. The method as recited in claim **5**, further comprising: facilitating a console for an administrator of a service provider to partition a storage device for creating the storage space according to the predefined capacity for the user of the wireless device, wherein the service provider provides services for the wireless device.

7. The system as recited in claim **2**, wherein the operation to the storage space comprises creating from the each of the wireless devices a folder in the storage space.

8. The system as recited in claim **2**, wherein the operation to the storage space comprises deleting or moving or copying or renaming, from the each of the wireless devices, a file or a folder being stored in the storage space.

9. The system as recited in claim **2**, wherein the link is wireless.

10. The system as recited in claim **9**, wherein the server is further configured to facilitate a console for an administrator of a service provider to partition a storage device for creating the storage space according to the predefined capacity for a user of one of the wireless devices, wherein the service provider provides services for the one of the wireless devices.

11. A non-transitory computer-readable storage medium comprising;

computer program instructions that, when executed by a server, configure the server to:

allocate a storage space of a predefined capacity for each of the wireless devices, create a file system for the storage space allocated for the each of the wireless devices;

establish a link for the each of the wireless devices access to the storage space; and

update the file system whenever a user of the each of the wireless devices performs an operation to the storage space, wherein the storage space being remotely located with respect to the each of the wireless devices,

wherein the updating of the file system comprises

updating the file system for storing a file therein, the storing of a file including to download a file from a remote web server, according to download information for the file cached in the each of the wireless devices received therefrom, directly into the storage device allocated thereto when the user via a web browser executed on the each of the wireless devices to perform an operation of downloading the file from the remote web server to the storage space instead of downloading the file into the each of the wireless devices itself.

12. The storage medium of claim **11**, wherein the program instructions further configure the server to facilitate a console for an administrator of a service provider to partition a storage device for creating the storage space for a user of one of the wireless devices, wherein the service provider provides services for the one of the wireless devices.

13. The storage medium of claim **11**, wherein the operation to the storage space further comprises deleting or moving or copying or renaming, from the each of the wireless devices, a file or a folder being stored in the storage space.

14. The storage medium of claim **11**, wherein the operation to the storage space comprise creating from the each of the wireless devices a folder in the storage space.

15. The method as recited in the claim **1**, wherein said for the wireless device access to the storage space comprises:

from the wireless device to retrieving a file from the storage device for access to a digital picture, a video, a music or a message being stored therein.

16. The system as recited in the claim **2**, wherein said for the wireless device access to the storage space comprises:

from the wireless device to retrieve a file from the storage device for access to a digital picture, a video, a music or a message being stored therein.

17. The storage medium of claim **11**, wherein said for the wireless device access to the storage space comprises:

from the wireless device to retrieve a file from the storage device for access to a digital picture, a video, a music or a message being stored therein.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 8,606,880 B2
APPLICATION NO. : 10/726897
DATED : December 10, 2013
INVENTOR(S) : Sheng Tai Ted Tsao

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Delete the title page and substitute therefore with the attached title page showing the corrected number of drawing sheets in patent.

In the Drawings:

FIG. 5 through FIG. 11C have been deleted.

In the Claims:

A) in column 6 line 16, column 6 line 43, column 7 line 36 - column 8 line 1, column 8 lines 22 – 23, column 8 lines 27 – 28, column 8 lines 32 – 33, please replace “storage device” with ---storage space---.

Signed and Sealed this
Twenty-fifth Day of February, 2014

Michelle K. Lee

Michelle K. Lee
Deputy Director of the United States Patent and Trademark Office

(12) **United States Patent**
Tsao

(10) **Patent No.: US 8,606,880 B2**
(45) **Date of Patent: Dec. 10, 2013**

(54) **USE OF WIRELESS DEVICES' EXTERNAL STORAGE**

(75) Inventor: **Sheng (Ted) Tai Tsao**, San Jose, CA (US)

(73) Assignee: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 2766 days.

(21) Appl. No.: **10/726,897**

(22) Filed: **Dec. 4, 2003**

(65) **Prior Publication Data**

US 2010/0005153 A1 Jan. 7, 2010

(51) **Int. Cl.**
G06F 15/16 (2006.01)

(52) **U.S. Cl.**
USPC **709/219**; 709/203; 709/226; 455/412.1; 455/899

(58) **Field of Classification Search**
USPC 709/200, 203, 217, 219, 226, 245; 455/412.1, 899
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

6,351,776 B1 * 2/2002 O'Brien et al. 709/245
6,356,941 B1 * 3/2002 Cohen 709/219
6,714,968 B1 * 3/2004 Prust 709/219
6,735,623 B1 * 5/2004 Prust 709/219
6,839,743 B2 * 1/2005 Shim 709/217
6,985,927 B2 * 1/2006 O'Brien et al. 709/213
7,506,034 B2 * 3/2009 Coates et al. 709/219
7,650,621 B2 * 1/2010 Thomas et al. 725/87
7,894,803 B2 * 2/2011 Kamada 455/414.3
2002/0059621 A1 * 5/2002 Thomas et al. 725/87
2002/0123336 A1 * 9/2002 Kamada 455/420
2002/0128036 A1 * 9/2002 Yach et al. 455/552
2002/0133597 A1 * 9/2002 Jhingan et al. 709/228
2003/0191716 A1 * 10/2003 Woods et al. 705/50

* cited by examiner

Primary Examiner — El Hadji Sall

(57) **ABSTRACT**

Adapting web-based external storage, wireless device can posses huge amount of storage that current any wireless device's internal storage can not provide. To effectively let the storage server providing external storage (file system) for wireless device, the storage of a storage server need to be partitioned into multiple small storage volume and need to be exported to each specific wireless device. The console support software coupled with web-server software of a server provides both users of wireless device and console through web-browser to perform tasks of creating and utilizing external storage (file system). To support larger number of wireless devices with external storage, a central controlled distributed scalable virtual machine infrastructure can be deployed. The larger number of storage server controlled by a central control system can satisfy unlimited wireless devices external storage needs.

17 Claims, 4 Drawing Sheets

Wireless out-band download

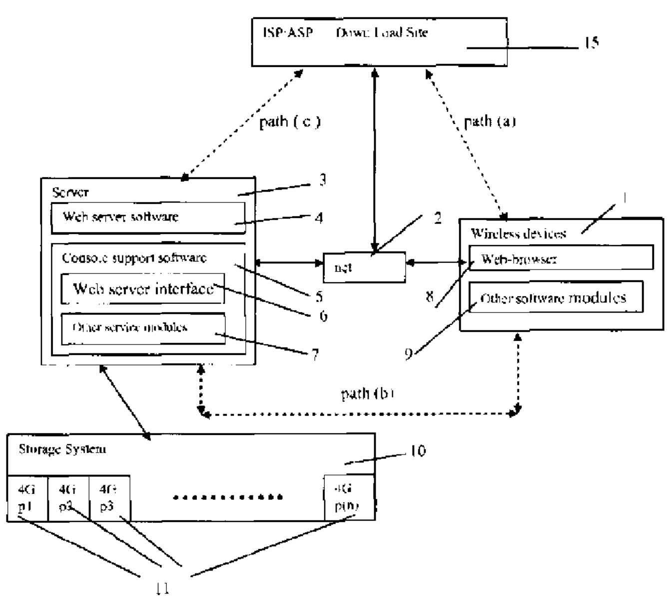


EXHIBIT D

US008856195B1

(12) **United States Patent**
Tsao

(10) **Patent No.: US 8,856,195 B1**
(45) **Date of Patent: *Oct. 7, 2014**

(54) **METHOD AND SYSTEM FOR WIRELESS DEVICE ACCESS TO EXTERNAL STORAGE**

(71) Applicant: **Sheng Tai Tsao**, Fremont, CA (US)

(72) Inventor: **Sheng Tai Tsao**, Fremont, CA (US)

(73) Assignee: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **14/153,052**

(22) Filed: **Jan. 12, 2014**

Related U.S. Application Data

(63) Continuation of application No. 14/079,831, filed on Nov. 14, 2013, which is a continuation of application No. 10/726,897, filed on Dec. 4, 2003, now Pat. No. 8,606,880.

(51) **Int. Cl.**
G06F 17/30 (2006.01)
H04W 72/04 (2009.01)

(52) **U.S. Cl.**
CPC ***G06F 17/30197*** (2013.01); ***H04W 72/04*** (2013.01)
USPC **707/821**; 707/827

(58) **Field of Classification Search**
USPC 707/821, 827
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,345,584	A *	9/1994	Hill	....	711/170
5,901,228	A *	5/1999	Crawford	....	705/34
5,969,716	A *	10/1999	Davis et al.	....	715/726
6,631,442	B1 *	10/2003	Blumenau	....	711/112
6,754,710	B1 *	6/2004	McAlear	....	709/227
7,054,903	B2 *	5/2006	Kadyk et al.	....	709/203

* cited by examiner

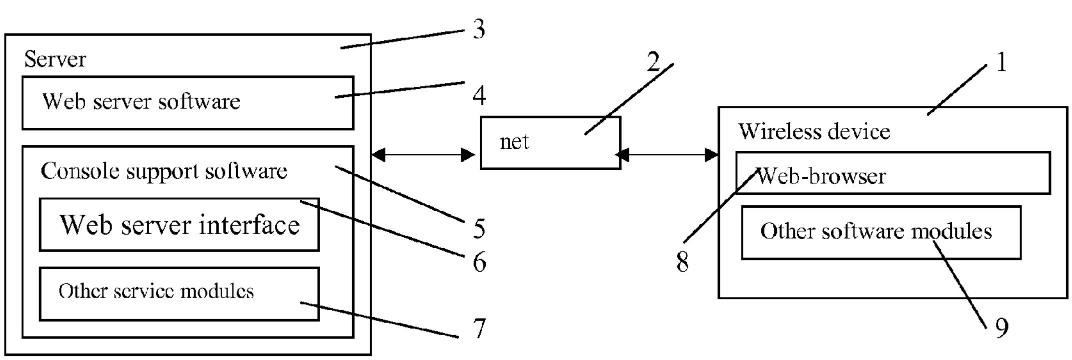
Primary Examiner — Kim Nguyen

(57) **ABSTRACT**

To meet the needs for storing larger volume personal information for user of wireless device, it is desire to provide extra storage space to the wireless device such as for cell phone or personal assistant device (PDA) etc due to the limited storage space that the wireless device has. Instant application disclosed a system and method for the wireless device to efficiently and effectively use remotely located storage space provided by a server for storing message or multimedia data such as videos, digital music, digital photo/picture.

20 Claims, 4 Drawing Sheets

Wireless devices supports in a simple environment



Wireless devices supports in a simple environment

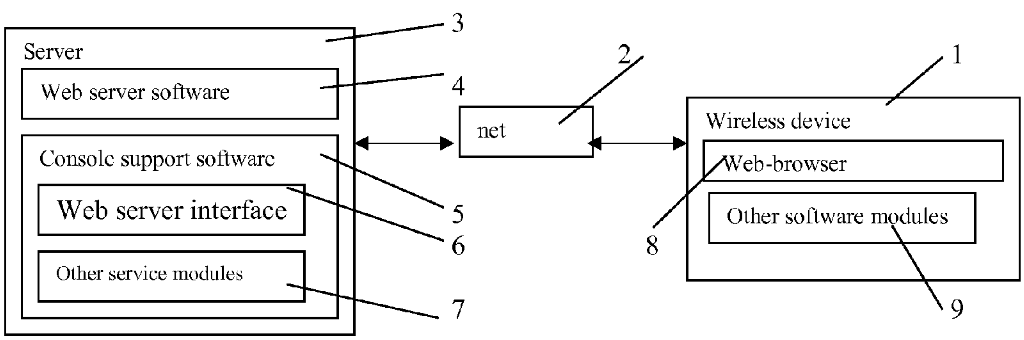


Fig. 1

Wireless devices access external storage through web browser

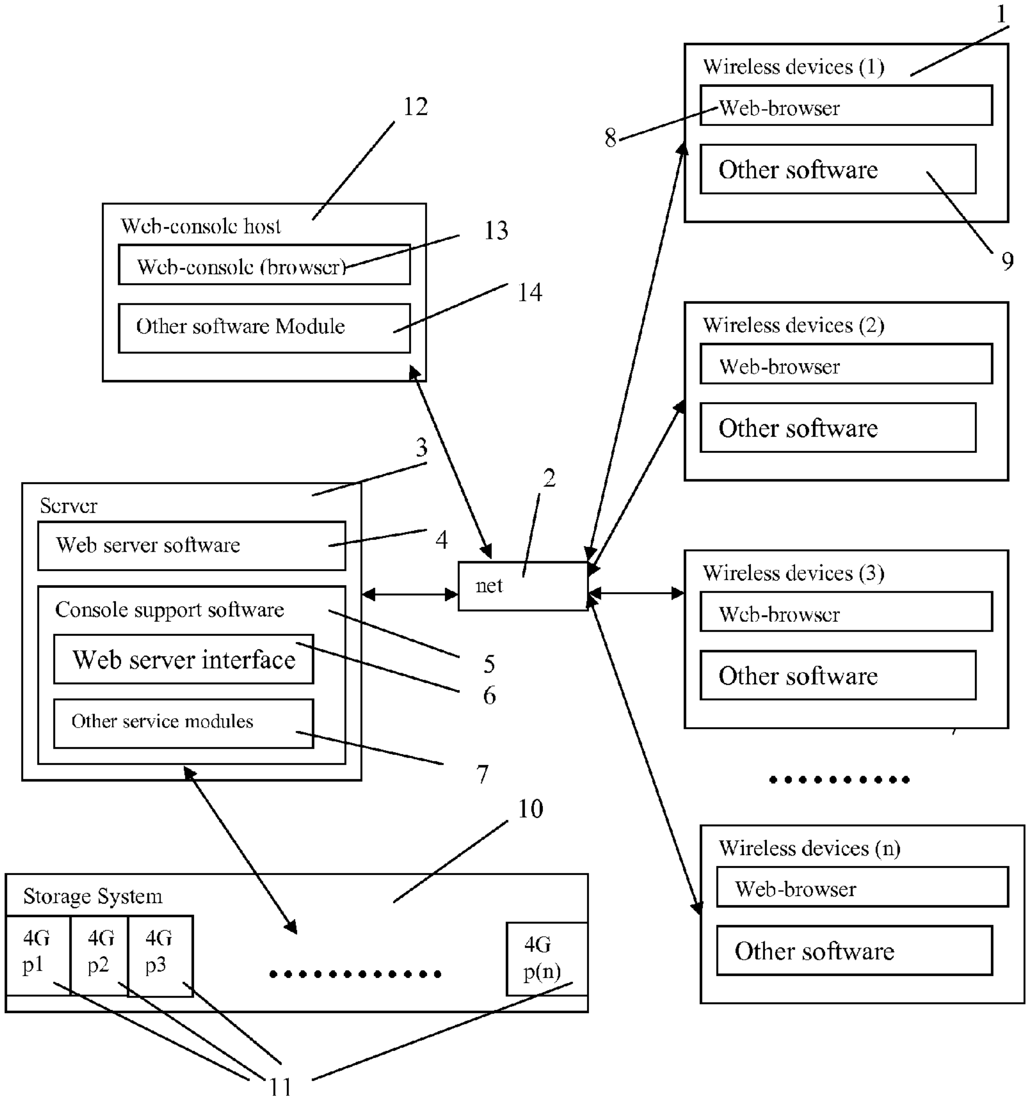


Fig. 2

Wireless out-band download

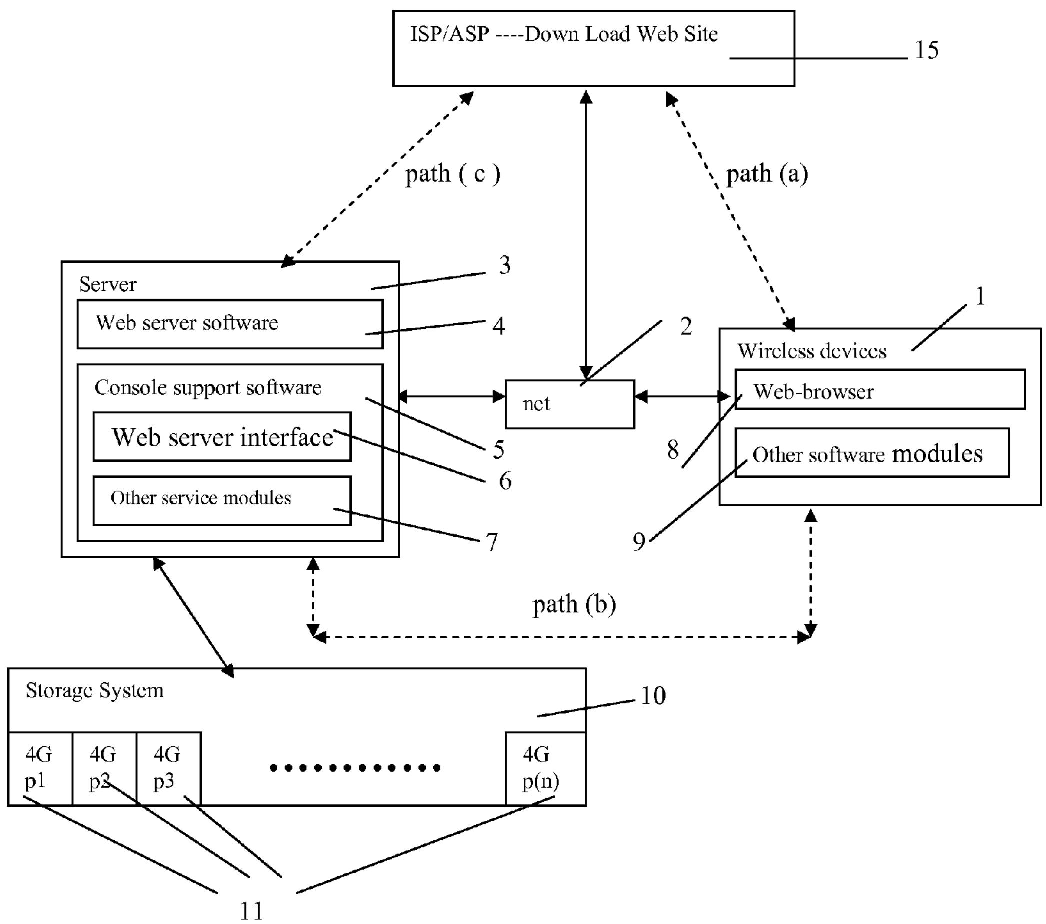


Fig. 3

The CCDSVM Support External Device for Huge Number of Wireless Device

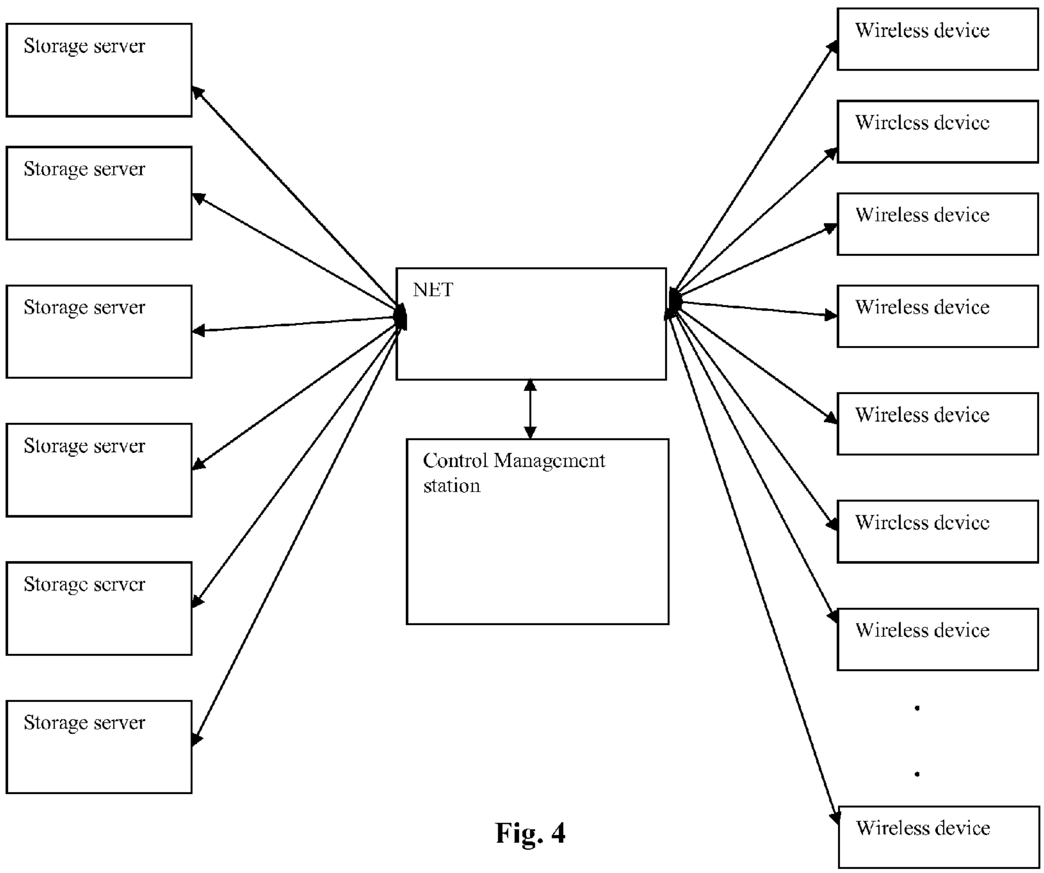


Fig. 4

METHOD AND SYSTEM FOR WIRELESS DEVICE ACCESS TO EXTERNAL STORAGE

CROSS REFERENCE TO PRIOR APPLICATION

This invention is a continuation application of the U.S. patent application Ser. No. 14/079,831 filed on Nov. 14, 2013, which in turn itself is a continuation application of U.S. patent application Ser. No. 10/726,897 filed on Dec. 4, 2003, which relates to a prior application No. 60/401,238 of "Concurrent Web Based Multi-task Support for Control Management System" filed on Aug. 6, 2002. This invention also relates to a prior application No. 60/402,626 of "IP Based Distributed Virtual SAN" filed on Aug. 12, 2002. All mentioned prior applications are herein referenced incorporated by reference in their entirety for all purpose.

FIELD OF THE INVENTION

This invention focuses on how a wireless device can actually use external storage provided by a storage server.

BACKGROUND INFORMATION

Terminology

The terminologies described in next few sections reflect the scope and are part of present invention.

The Internal Storage of a System:

The storage media such as hard disk drives, memory sticks, memory etc. is connected to a system directly through bus or a few inches of cable. Therefore, the storage media actually is a component of the system in an enclosure.

The External Storage of a System:

The external storage media is not a component of the system in a same enclosure. Therefore, they have to be connected through a connecting medium (e.g. a cable) such as Ethernet cable for IP based storage, Fiber channel cable for fiber channel storage, or such as wireless medium and etc. The storage media of an external storage could be magnetic hard disk drives, solid state disk, optical storage drives, memory card, etc. and could be in any form such as Raid, which usually consists of a group of hard disk drives.

The Storage Partition, its Volumes, and the Corresponding File System:

To effectively use storage system, each storage device usually needs to be partitioned into small volumes. After the partition, each of the volumes can be used to establish a file system on it. To simplify the discussion herein, the term of the storage volume, its corresponding file system, and the term of the partition of the storage device are often used without differentiation.

CCDSVM:

It is an abbreviation for a central controlled distributed scalable virtual machine system. The CCDSVM allows a control management station to control a group of systems and provide distributed services to a client system on the Internet, the Intranet, and an LAN environment.

ISP & ASP:

The ISP refers to Internet service provider and the ASP refers to application service provider.

FIGURES

FIG. **1** illustrates an embodiment of the instant application, the FIG. **1** is the same as FIG. **1** of the prior application of the "Concurrent Web Based Multi-task Support for Control Management System" with an exception of replacing a console host with a wireless device.

FIG. **2** is the same as FIG. **1** of the above except that it shows a more detailed storage system controlled by a server. In addition, multiple wireless devices are presented for access to the storage system.

FIG. **3** shows a scheme of a wireless device downloading contents from an ISP/ASP or other web sites to an external storage allocated for the wireless device.

FIG. **4** similar to the FIG. **1** of the prior application of the "IP Based Distributed Virtual SAN" with exception that each IP storage server provides a file system as external storage for each of the wireless devices instead of providing IP based virtual SAN service. Also, each host in said FIG. **1** actually is replaced by a wireless device of FIG. **4**.

Unless specified, the programming languages and the protocols used by each software modules of instant application, and the computing systems used in this invention are assumed to be the same as described in the mentioned prior patent applications.

In addition, in the drawing, like elements are designated by like reference numbers. Further, when a list of identical elements is present, only one element will be given the reference number.

BRIEF DESCRIPTION OF THE INVENTION

Today the wireless users commonly face a problem of lack of storage capacity configured on their wireless devices such as cell phone or PDA, which are usually limited to 256 MB for PDA and much less for cell phone. To effectively solve this problem and let users own multiple gigabytes (GB) of storage for their wireless devices as well as allowing the users to use the GB storage for their multimedia applications, the storage of a server can be used as the external storage for the wireless devices. This technology has been briefly introduced in the prior parent applications.

Now let us examine how the external storage can actually be used by the wireless devices. First, let each server unit (e.g. the server **3** of the FIG. **2**) partitions its storage system into volume and each of the volumes will have multiple GB in size. Therefore, each user of the wireless devices can be exclusively assigned and access a specific storage volume. For example, if we need to provide each user a 4 GB storage space, then a 160 GB disk drive can support 40 users. Therefore, a 4096 GB storage system on the server unit can support a total of 1024 wireless devices for users. Further, any data on the wireless device can be transmitted to an assigned storage volume. In addition, the user of the wireless device also can download the multimedia data from an ISP or ASP to the assigned storage volume of a designated server unit through out-band approach shown in FIG. **3**. Finally, the user can use a web-browser, which has a functionality of invoking embedded video or music, to enjoy his/her stored multimedia contents.

These and other futures, aspects and advantages of the present invention will become understood with reference to the following description, appended claims, and accompanying figures.

DESCRIPTION OF THE DRAWINGS

Referring now to FIG. **1**, it demonstrates a configuration comprising a network connecting a wireless device and a server.

In the FIG. **1**, Net (**2**) represents a communication link, which may be combined with wireless and wired connection media and guarantee that the communication packets can be sent/received between the wireless device and the server. It is also assumed that the net (**2**) representing an communication infrastructure is built up in such way that a user of a wireless device can access and browse any web-site on the Internet, the Intranet, or a local area network (LAN).

In FIG. **1**, the console support software (**5**) on the server (**3**) can be configured to support web-based multi-tasks for the user of the wireless device (**1**) via a web browser **8**. Further, the user of the wireless device is able to perform creating structured layered files directories or folders, and perform data management operations, such as delete, move, copy, rename for data files or folders/directories and etc. on an assigned storage volume associated with the server (**3**).

In addition, the other software modules (**9**) of the wireless device (**1**) is configured to send data to or receive data from the other service modules (**7**) running on the server (**3**) via communication link (**2**) through a suitable IP or non-IP based protocol. The data being sent cold be a digital photo picture, a message and etc.

Also, the console supporting software (**5**) of the server (**3**) and the other software modules (**9**) of the wireless device (**1**) can be implemented with any suitable languages such as C, C++, Java, etc. depending on the implementation.

Besides, the web-browser (**8**) of the wireless device (**1**) can be implemented any suitable software. The web browser is configured to communication with web server software (**4**) on the server (**3**) with any other web server through the HTTP protocol.

On the other hand, FIG. **2** has demonstrated that the storage system (**10**) of a server **3** can be allocated to multiple wireless devices.

First, the storage system (**10**) of the server (**3**) can be partitioned into multiple storage volumes (**11**) by administration staff through a web-console (**13**) of a console host (**12**).

Second, the storage system (**10**) of the server (**3**) can be partitioned in such way that each of the wireless devices can be allocated with a storage volume having a desired size, therefore, the server **3** can support maximum numbers of the wireless devices.

In addition, the storage connection media could be any kind of cables, such as SCSI cable, IP cable, Fiber cable etc. or could be wireless communication media. The storage system itself could be various types.

Finally, the storage system (**10**) can be accessed by each of the wireless devices through IP or non-IP based network and protocols.

FIG. **3** has demonstrated that a user from a web-browser (**8**) on a wireless device (**1**) can download data from a known web-site (**15**) to his/her allocated external storage (**10**) on the server (**3**). The dash-lined path (a) represents a communication channel between the wireless device (**1**) and a remote downloading web-site (**15**) that provides downloading contents. The dash-lined path (b) represents a communication channel between the wireless devices (**1**) and the storage server (**3**). The dash-lined path (c) represents a communication channel between the server **3** and the remote web-server (**15**).

THE DETAILED DESCRIPTION OF THE INVENTION

The Use of the External Storage by the Wireless Device:

The FIG. **2** shows a simplified diagram of the wireless devices (**1**) using the external storage system (**10**) of the server (**3**) for effectively resolving the storage limitation problem for the wireless devices (**1**).

Partition Storage Volumes (FIG. **2**)

With this invention, the entire storage (**10**) on the server (**3**) needs to be partitioned into suitable size of volumes (**11**) such as 4 GB for each volume. This will allow the server **3** to serve maximum number of the wireless devices (**1**). With the web console support software (**5**) of the server (**3**), tasks of partitioning the storage system **10** can be done through a web-console (**13**) on a console host (**12**) by an administrative staff.

In order to support storage partitioning, first the console support software (**5**) of the server (**3**) must send storage information of the server (**3**) to the web-console (**13**) of the console host (**12**). The storage information includes each storage device's name and total size etc. Second, based on the received storage information the administration staff on the console host (**12**) can use a web-console (**13**) to fill out a request and send the request for storage partition information to the console support software (**5**) of the server (**3**). The storage partition information includes the number of the partitions (volumes) and the size of each partition (volume). Third, upon receiving the request of storage partition information from the web-console (**13** of FIG. **2**) of the console host (**12**), the console support software (**5**) of the server (**3**) performs the actual storage partition to divide the entire storage into multiple small volumes. Finally, for each small storage volume, a corresponding file system could be built up.

Assign Storage Volumes (FIG. **2**):

Each of the storage volumes (**11**) together with its corresponding file system on the storage system (**10**) of the server (**3**) needs to be exclusively allocated and exported to a specific wireless device (**1**) by the console support software (**5**) of the server (**3**).

Data and Storage Volume Management (FIG. **2**)

With the support of the console support software modules (**5**) of the server (**3**), the user of the wireless device (**1**) can via the web-browser **8** of FIG. **2** setup folder/directory structure on the file system of his/her assigned external storage volume (**11**). In addition, the user of the wireless device (**1**) can via the web-browser **8** of FIG. **2** perform all data management operations such as delete, copy, move, rename etc. for the file system.

In order to support such data management on the external storage (**10**) allocated to the wireless device (**1** of FIG. **2**) via the web-browser **8** of FIG. **2**, first the console support software modules (**5**) of the server (**3**) must communicate with the web-browser (**8**) of the wireless device (**1**). Therefore, the user from the web-browser (**8**) of the wireless device (**1**) can choose desired data management operations and send operation information to the console support software modules (**5**) of the server (**3**). The mentioned operations include establishing folder/directory, copying, moving, or reaming data file etc. Second, upon receiving the data management operation, the console support software modules (**5**) of the server (**3**) actually processes/executes these requested operations for the assigned file system of an allocated storage volume (**11**) on the server (**3**).

Store Data from Wireless Device into External Storage (FIG. **2**)

To store the data such as digital photo pictures or messages into the file system on the allocated storage (**10**) of the server (**3**), the other software modules (**9**) of the wireless device (**1**) need to send these data to the other service modules (**7**) of the server (**3**) via communication link between them. Upon receiving the data, the other service modules (**7**) of the server (**3**) write these data to the file system of the allocated storage volume (**11**) for the wireless device **1**. The protocol used between these two communication entities could be either IP or non-IP based protocol.

Download Data from a Remote Web Server Site into Allocated Storage Volume

Now, referring to FIG. **3**, If a user of the wireless device (**1**) wants to download a data from a remote web server (**15**) into the allocated file system on the allocated storage volume (**11**) of the external storage system (**10**) on the server (**3**), the following steps are required:

1) The user of the wireless device (**1**) via a web-browser (**8**) access to a remote downloading web server site (**15**) to obtain information for the downloading via path (a) of FIG. **3**. For example, the user access to a web-page which contains the data name for the downloading.

2) The other software modules (**9**) of the wireless device (**1**) obtain the downloading information, which becomes available in the cached web-pages on the wireless device (**1**) after the web-browser (**8**) access to the web site (**15**).

3) The other software modules (**9**) of the wireless device (**1**) send the obtained downloading information to other service modules (**7**) of the storage server (**3**) via path (b) of FIG. **3**.

4) Upon receiving the downloading information from the wireless device (**1**), the other service module (**7**) of the storage server (**3**) sends a web download request to the web-site (**15**) via path (c) of FIG. **3** and receives the downloading data streams from the web server of the web-site (**15**).

5) Upon receiving downloading data streams, the other service modules (**7**) of the storage server (**3**) write the data streams into the allocated file system on the allocated storage volume (**11**) on the server (**3**) for the wireless device (**1**).

Retrieve Data from Allocated Storage for the Wireless Device

1) If a web-browser (**8**) on a wireless device **1** has embedded video or music functionality, a user of the wireless device (**1**) can use the browser to retrieve and play multimedia data file such as video or music stored in the allocated storage volume (**10**) located on the server (**3**).

2) If there is a needs, the other software module (**9**) of the wireless device (**1**) also can retrieve data file from the allocated file system of the allocated storage volume (**11**) located on the server (**3**).

Support External Storage for a Large Number of the Wireless Devices

If there is a need to provide each user a 2 GB of storage space, then a 160 GB disk drive can support 80 users. A 4096 GB (4 Tera Bytes) storage system on the server unit can support 2024 user. Each of the server units only can efficiently support a limited size of the storage system. In order to support a large number of the wireless devices, such as for 500,000 wireless devices, a larger number of the servers is required, in this case 250 servers is required. In order to let a larger number of the servers to effectively support the larger number of the wireless devices, an infrastructure like CCDSVM is desirable, which has been described in previous patent applications. With the CCDSVM the control management station can control larger number of storage servers to provide external storage for a huge number of the wireless devices.

The invention claimed is:

1. A method for expanding storage capacity for a wireless device, the method comprising:

allocating via a server a storage space of a predefined capacity for the wireless device, the storage space being remotely located with respect to the wireless device;

creating a file system for the storage space allocated for the wireless device;

establishing a wireless link for the wireless device access to the storage space; and

updating the file system whenever a user of the wireless device performs an operation to the storage space,

wherein the updating of the file system includes updating the file system in response to an operation performed by the user for storing a file therein, the storing of a file including to download a file from a remote web server directly into the storage space.

2. The method as recited in claim **1**, wherein the operation to the storage space comprises creating, from the wireless device, a folder in a folder structure configured in the storage space.

3. The method as recited in claim **2**, wherein the wireless device is configured to allow the user thereof access to the storage space for storing therein or retrieving therefrom a message or multimedia data of video, digital music, or digital picture.

4. The method as recited in claim **2**, wherein the operation to the storage space further comprises deleting or moving or copying or renaming, from the wireless device, a file or a folder stored in the storage space.

5. The method as recited in claim **1**, wherein said downloading a file further comprises:

obtaining downloading information for the file;

transmitting the downloading information cached in the wireless device to the server; and

causing the server in accordance with the downloading information to download the file directly into the storage space.

6. A system for expanding storage capacity for a plurality of wireless devices, the system comprising:

the wireless devices; and

a server configured to:

allocate a storage space of a predefined capacity for each of the wireless devices, create a file system for the storage space allocated for the each of the wireless devices;

establish a wireless link for the each of the wireless devices access to the storage space; and

update the file system whenever a user of the each of the wireless devices performs an operation to the storage space, wherein the storage space being remotely located with respect to the each of the wireless devices,

wherein the updating of the file system includes updating the file system in response to an operation performed by the user for storing a file therein, the storing of a file including to download a file from a remote web server directly into the storage space; and

wherein each of the wireless devices is operable access to the storage space allocated to the each of the wireless devices.

7. The system as recited in claim **6**, wherein the operation to the storage space comprises creating, from the each of the wireless devices, a folder in a folder structure configured in the storage space allocated to the each of the wireless devices.

8. The system as recited in claim **7**, wherein the each of the wireless devices is configured to allow the user thereof access to the folder structure for storing therein or retrieving therefrom a message or multimedia data of video, digital music, or digital picture.

9. The system as recited in claim **7**, wherein the operation to the storage space further comprises deleting or moving or copying or renaming, from the each of the wireless devices, a file or a folder in the storage space.

10. The system as recited in claim **7**, wherein the each of the wireless devices further is operable to execute a web browser to display information about files and folders, in the storage space, through which said operation to the storage space is performed.

11. The system as recited in claim **6**, wherein said downloading a file further comprises:

instructing the each of the wireless devices to obtain downloading information for the file,

transmit the downloading information cached in the each of the wireless devices to the server, and

cause the server in accordance with the downloading information to download the file directly into the storage space.

12. The system as recited in claim **6**, wherein said creating a file system for the storage space comprises partitioning a storage device to create the storage space according to the predefined capacity.

13. The system as recited in claim **12**, wherein the server further facilitates a console for an administrator of a service provider to allocate the storage space for a subscriber of one of the wireless devices, wherein the service provider provides services for the one of the wireless devices.

14. The system as recited in claim **12**, wherein each of the wireless devices further has a function of making or receiving a phone call.

15. A non-transitory computer-readable medium comprising:

program instructions that, when executed by a server, cause the server to:

allocate a storage space of a predefined capacity for each of wireless devices, create a file system for the storage space allocated for the each of the wireless devices;

establish a wireless link for the each of the wireless devices access to the storage space; and

update the file system whenever a user of the each of the wireless devices performs an operation to the storage space, wherein the storage space being remotely located with respect to the each of the wireless devices,

wherein the updating of the file system includes updating the file system in response to an operation performed by the user for storing a file therein, the storing of a file including to download a file from a remote web server directly into the storage space.

16. The computer-readable medium of claim **15**, wherein the program instructions cause the server to update the file system whenever receiving an operation, from the each of the wireless devices, for creating a folder in a folder structure configured in the storage space.

17. The computer-readable medium of claim **16**, wherein the program instructions cause the server to update the file system whenever receiving an operation, from the each of the wireless devices, for access to the folder for storing therein a message or multimedia data of video, digital music or digital picture.

18. The computer-readable medium of claim **17**, wherein the program instructions cause the server to allow the user of the each of the wireless devices to retrieve from the storage space a message or multimedia data of video or digital music or digital picture across world wide web.

19. The computer-readable medium of claim **16**, wherein the program instructions cause the server to update the file system whenever receiving an operation, from the each of the wireless devices, for deleting or moving or copying or renaming a file or a folder in the storage space.

20. The computer-readable medium of claim **15**, wherein said downloading a file further comprises:

causing the each of the wireless devices to obtain downloading information for the file;

transmitting the downloading information cached in the each of the wireless devices to the server; and

causing the server in accordance with the downloading information to download the file directly into the storage space.

* * * * *

EXHIBIT E

US008868690B2

(12) **United States Patent**
Tsao

(10) **Patent No.:** **US 8,868,690 B2**
(45) **Date of Patent:** ***Oct. 21, 2014**

(54) **SYSTEM AND METHOD FOR SUPPORT WIRELESS DEVICE ACCESS TO EXTERNAL STORAGE**

(71) Applicant: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(72) Inventor: **Sheng Tai Tsao**, Fremont, CA (US)

(73) Assignee: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **14/079,831**

(22) Filed: **Nov. 14, 2013**

(65) **Prior Publication Data**

US 2014/0089360 A1 Mar. 27, 2014

Related U.S. Application Data

(63) Continuation of application No. 10/726,897, filed on Dec. 4, 2003, now Pat. No. 8,606,880.

(51) **Int. Cl.**

G06F 15/16	(2006.01)
G06F 17/30	(2006.01)
H04L 29/08	(2006.01)

(52) **U.S. Cl.**
CPC ***G06F 17/30194*** (2013.01); ***G06F 17/30067*** (2013.01); ***H04L 67/06*** (2013.01); ***H04L 67/1097*** (2013.01); ***H04L 67/02*** (2013.01); ***H04L 67/2842*** (2013.01)
USPC **709/219**; 709/203; 455/899

(58) **Field of Classification Search**
USPC 709/219, 203, 226, 220, 200; 455/412.1, 899
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

2002/0133597	A1 *	9/2002	Jhingan et al.	709/228
2003/0072031	A1 *	4/2003	Kuwata et al.	358/1.15
2004/0186750	A1 *	9/2004	Surbey et al.	705/4

* cited by examiner

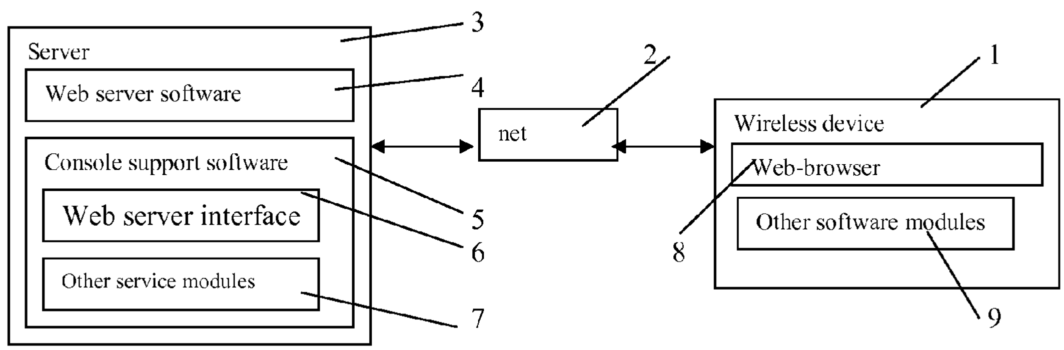
Primary Examiner — El Hadji Sall

(57) **ABSTRACT**

To meet the needs for storing larger volume personal information for user of wireless device, it is desire to provide extra storage space to the wireless device such as for cell phone etc due to the limited storage space that the wireless device has. Instant application disclosed a system and method for the wireless device to efficiently and effectively use remotely located storage space provided by a server.

20 Claims, 4 Drawing Sheets

Wireless devices supports in a simple environment



Wireless devices supports in a simple environment

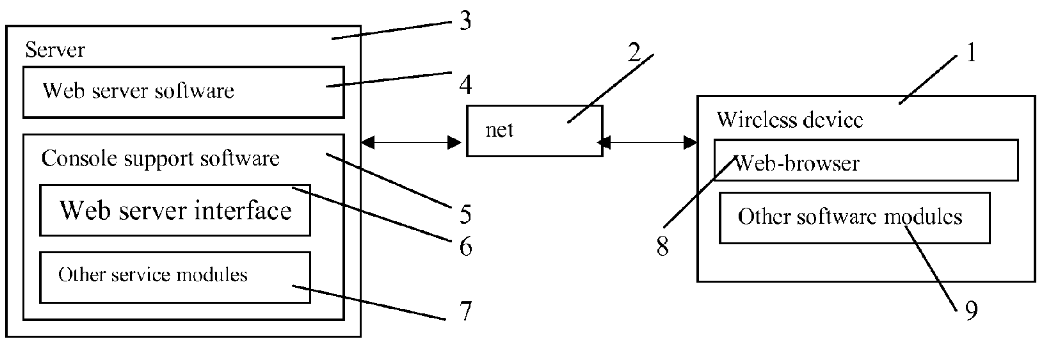


Fig. 1

Wireless devices access external storage through web browser

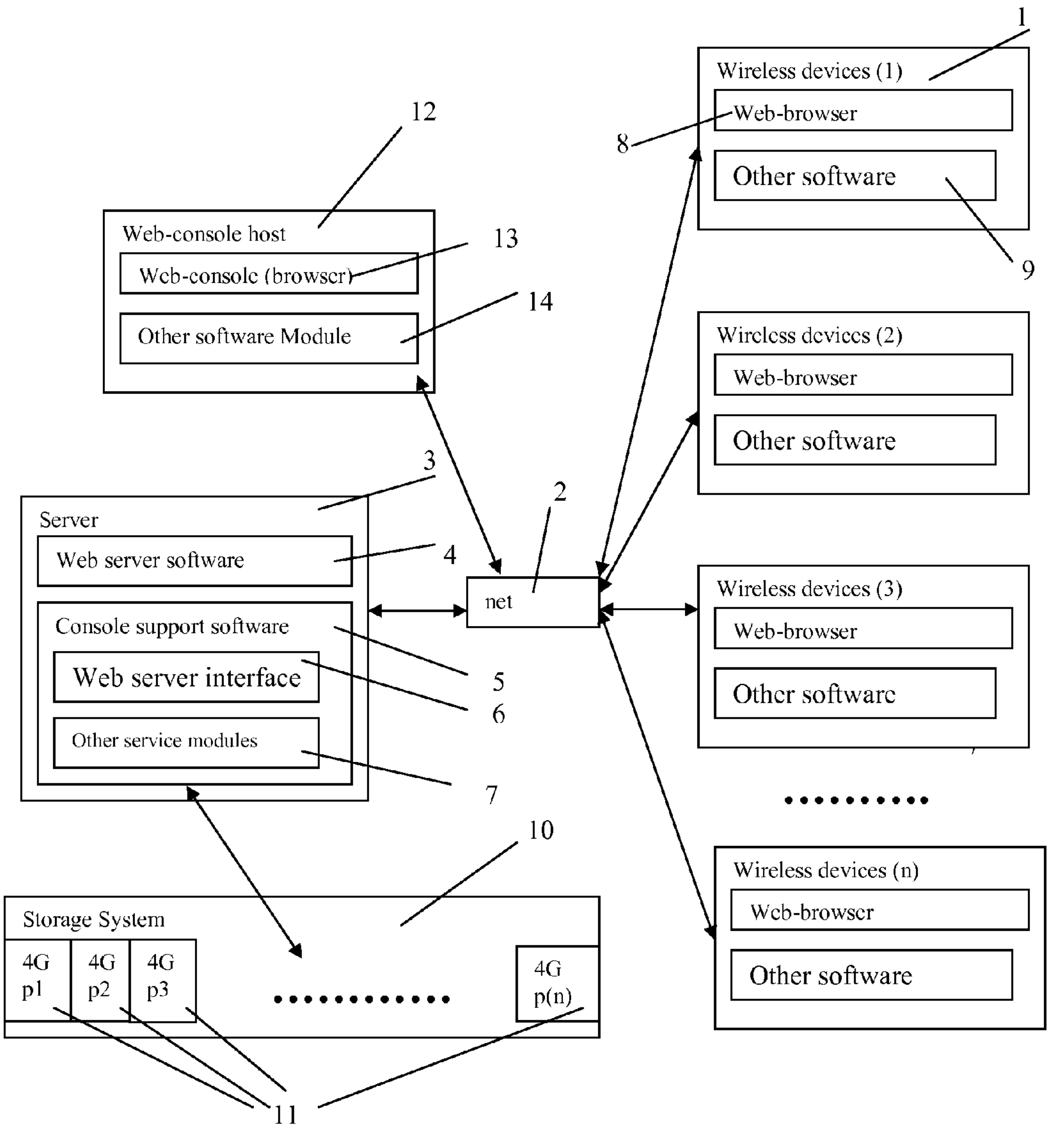


Fig. 2

Wireless out-band download

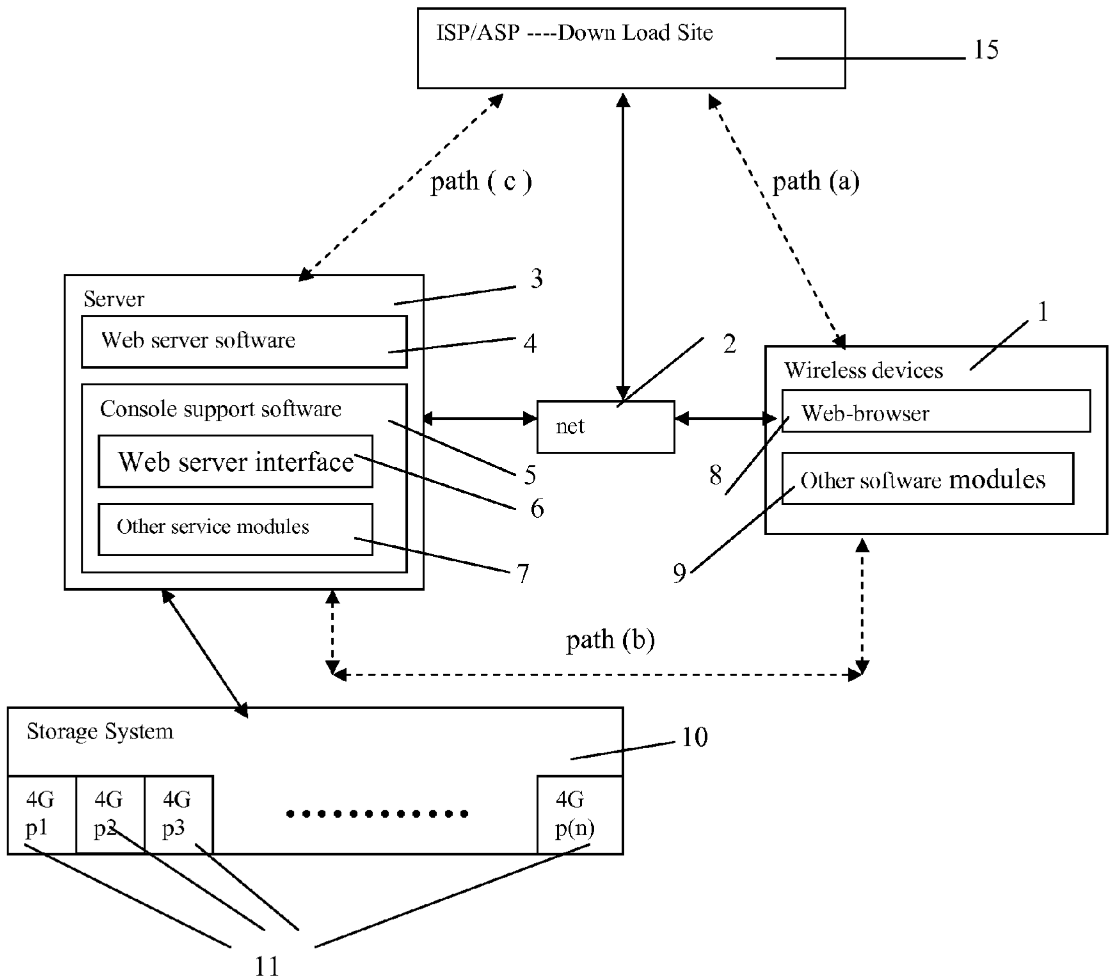


Fig. 3

The CCDSVM Support External Device for Huge Number of Wireless Device

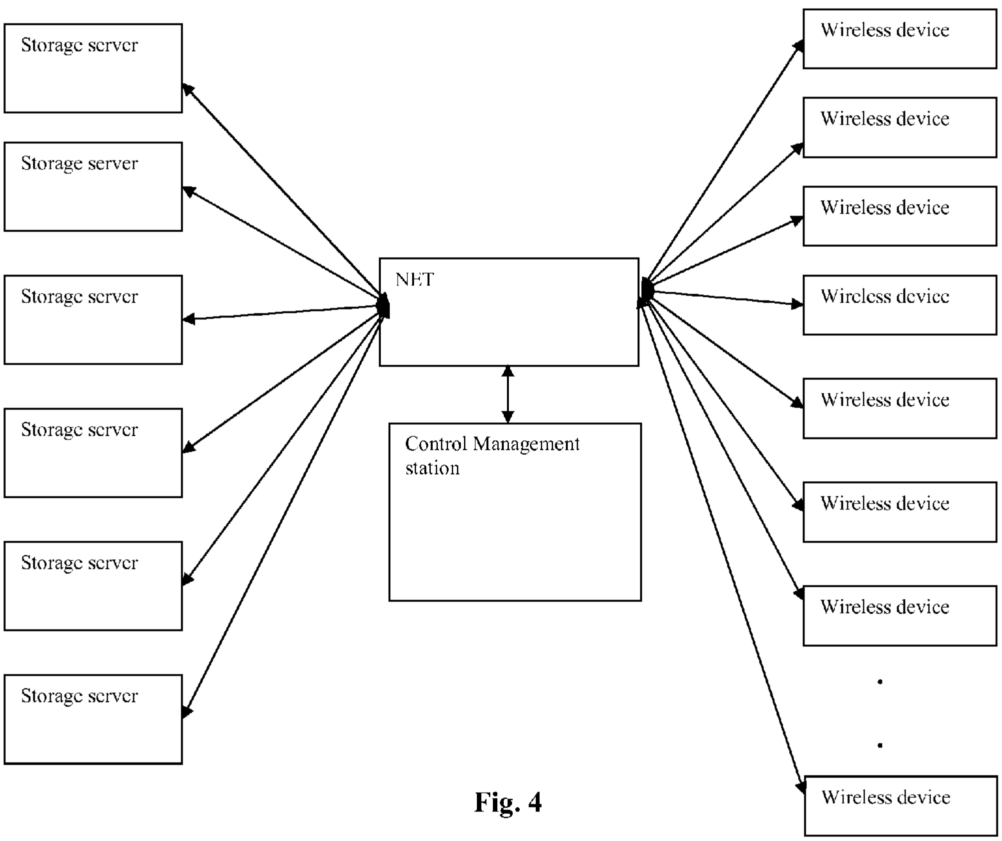


Fig. 4

SYSTEM AND METHOD FOR SUPPORT WIRELESS DEVICE ACCESS TO EXTERNAL STORAGE

CROSS REFERENCE TO PRIOR APPLICATION

This invention is a continuation application of the U.S. patent application Ser. No. 10/726,897 filed on Dec. 4, 2003, which relates to a provisional application No. 60/401,238 of "Concurrent Web Based Multi-task Support for Control Management System" filed on Aug. 6, 2002. This invention also relates to a provisional application No. 60/402,626 of "IP Based Distributed Virtual SAN" filed on Aug. 12, 2002. All mentioned prior applications are herein referenced incorporated by reference in their entirety for all purpose.

FIELD OF THE INVENTION

This invention focuses on how a wireless device can actually use external storage provided by a storage server.

BACKGROUND INFORMATION

Terminology:

The terminologies described in next few sections reflect the scope and are part of present invention.

The Internal Storage of a System:

The storage media such as hard disk drives, memory sticks, memory etc. is connected to a system directly through bus or a few inches of cable. Therefore, the storage media actually is a component of the system in an enclosure.

The External Storage of a System:

The external storage media is not a component of the system in a same enclosure. Therefore, they have to be connected through a connecting medium (e.g. a cable) such as Ethernet cable for IP based storage, Fiber channel cable for fiber channel storage, or such as wireless medium and etc. The storage media of an external storage could be magnetic hard disk drives, solid state disk, optical storage drives, memory card, etc. and could be in any form such as Raid, which usually consists of a group of hard disk drives.

The Storage Partition, its Volumes, and the Corresponding File System:

To effectively use storage system, each storage device usually needs to be partitioned into small volumes. After the partition, each of the volumes can be used to establish a file system on it. To simplify the discussion herein, the term of the storage volume, its corresponding file system, and the term of the partition of the storage device are often used without differentiation.

CCDSVM:

It is an abbreviation for a central controlled distributed scalable virtual machine system. The CCDSVM allows a control management station to control a group of systems and provide distributed services to a client system on the Internet, the Intranet, and an LAN environment.

ISP & ASP:

The ISP refers to Internet service provider and the ASP refers to application service provider.

FIGURES

FIG. **1** illustrates an embodiment of the instant application, the FIG. **1** is the same as FIG. **1** of the previous application of the "Concurrent Web Based Multi-task Support for Control Management System" with an exception of replacing a console host with a wireless device.

FIG. **2** is the same as FIG. **1** of the above except that it shows a more detailed storage system controlled by a server. In addition, multiple wireless devices are presented for access to the storage system.

FIG. **3** shows a scheme of a wireless device downloading contents from an ISP/ASP or other web sites to an external storage allocated for the wireless device.

FIG. **4** similar to the FIG. **1** of the previous application of the "IP Based Distributed Virtual SAN" with exception that each IP storage server provides a file system as external storage for each of the wireless devices instead of providing IP based virtual SAN service. Also, each host in said FIG. **1** actually is replaced by a wireless device of FIG. **4**.

Unless specified, the programming languages and the protocols used by each software modules of instant application, and the computing systems used in this invention are assumed to be the same as described in the mentioned previous patent applications.

In addition, in the drawing, like elements are designated by like reference numbers. Further, when a list of identical elements is present, only one element will be given the reference number.

BRIEF DESCRIPTION OF THE INVENTION

Today the wireless users commonly face a problem of lack of storage capacity configured on their wireless devices such as cell phone or PDA, which are usually limited to 256 MB for PDA and much less for cell phone. To effectively solve this problem and let users own multiple gigabytes (GB) of storage for their wireless devices as well as allowing the users to use the GB storage for their multimedia applications, the storage of a server can be used as the external storage for the wireless devices. This technology has been briefly introduced in the previous parent patent applications.

Now let us examine how the external storage can actually be used by the wireless devices. First, let each server unit (e.g. the server **3** of the FIG. **2**) partitions its storage system into volume and each of the volumes will have multiple GB in size. Therefore, each user of the wireless devices can be exclusively assigned and access a specific storage volume. For example, if we need to provide each user a 4 GB storage space, then a 160 GB disk drive can support 40 users. Therefore, a 4096 GB storage system on the server unit can support a total of 1024 wireless devices for users. Further, any data on the wireless device can be transmitted to an assigned storage volume. In addition, the user of the wireless device also can download the multimedia data from an ISP or ASP to the assigned storage volume of a designated server unit through out-band approach shown in FIG. **3**. Finally, the user can use a web-browser, which has a functionality of invoking embedded video or music, to enjoy his/her stored multimedia contents.

These and other futures, aspects and advantages of the present invention will become understood with reference to the following description, appended claims, and accompanying figures.

DESCRIPTION OF THE DRAWINGS

Referring now to FIG. **1**, it demonstrates a configuration comprising a network connecting a wireless device and a server.

In the FIG. **1**, Net (**2**) represents a communication link, which may be combined with wireless and wired connection media and guarantee that the communication packets can be sent/received between the wireless device and the server. It is also assumed that the net (**2**) representing an communication infrastructure is built up in such way that a user of a wireless device can access and browse any web-site on the Internet, the Intranet, or a local area network (LAN).

In FIG. **1**, the console support software (**5**) on the server (**3**) can be configured to support web-based multi-tasks for the user of the wireless device (**1**) via a web browser **8**. Further, the user of the wireless device is able to perform creating structured layered files directories or folders, and perform data management operations, such as delete, move, copy, rename for data files or folders/directories and etc. on an assigned storage volume associated with the server (**3**).

In addition, the other software modules (**9**) of the wireless device (**1**) is configured to send data to or receive data from the other service modules (**7**) running on the server (**3**) via communication link (**2**) through a suitable IP or non-IP based protocol. The data being sent cold be a digital photo picture, a message and etc.

Also, the console supporting software (**5**) of the server (**3**) and the other software modules (**9**) of the wireless device (**1**) can be implemented with any suitable languages such as C, C++, Java, etc. depending on the implementation.

Besides, the web-browser (**8**) of the wireless device (**1**) can be implemented any suitable software. The web browser is configured to communication with web server software (**4**) on the server (**3**) with any other web server through the HTTP protocol.

On the other hand, FIG. **2** has demonstrated that the storage system **10** of a server **3** can be allocated to multiple wireless devices.

First, the storage system (**10**) of the server (**3**) can be partitioned into multiple storage volumes (**11**) by administration staff through a web-console (**13**) of a console host (**12**).

Second, the storage system (**10**) of the server (**3**) can be partitioned in such way that each of the wireless devices can be allocated with a storage volume having a desired size, therefore, the server **3** can support maximum numbers of the wireless devices.

In addition, the storage connection media could be any kind of cables, such as SCSI cable, IP cable, Fiber cable etc. or could be wireless communication media. The storage system itself could be various types.

Finally, the storage system **10** can be accessed by each of the wireless devices through IP or non-IP based network and protocols.

FIG. **3** has demonstrated that a user from a web-browser (**8**) on a wireless device (**1**) can download data from a known web-site (**15**) to his/her allocated external storage (**10**) on the server (**3**). The dash-lined path (a) represents a communication channel between the wireless device (**1**) and a remote downloading web-site (**15**) that provides downloading contents. The dash-lined path (b) represents a communication channel between the wireless devices (**1**) and the storage server (**3**). The dash-lined path (c) represents a communication channel between the server **3** and the remote web-server (**15**).

THE DETAILED DESCRIPTION OF THE INVENTION

The Use of the External Storage by the Wireless Device:

The FIG. **2** shows a simplified diagram of the wireless devices (**1**) using the external storage system (**10**) of the server (**3**) for effectively resolving the storage limitation problem for the wireless devices (**1**).

Partition Storage Volumes (FIG. **2**)

With this invention, the entire storage (**10**) on the server (**3**) needs to be partitioned into suitable size of volumes (**11**) such as 4 GB for each volume. This will allow the server **3** to serve maximum number of the wireless devices (**1**). With the web console support software (**5**) of the server (**3**), tasks of partitioning the storage system **10** can be done through a web-console (**13**) on a console host (**12**) by an administrative staff.

In order to support storage partitioning, first the console support software (**5**) of the server (**3**) must send storage information of the server (**3**) to the web-console (**13**) of the console host (**12**). The storage information includes each storage device's name and total size etc. Second, based on the received storage information the administration staff on the console host (**12**) can use a web-console (**13**) to fill out a request and send the request for storage partition information to the console support software (**5**) of the server (**3**). The storage partition information includes the number of the partitions (volumes) and the size of each partition (volume). Third, upon receiving the request of storage partition information from the web-console (**13** of FIG. **2**) of the console host (**12**), the console support software (**5**) of the server (**3**) performs the actual storage partition to divide the entire storage into multiple small volumes. Finally, for each small storage volume, a corresponding file system could be built up.

Assign Storage Volumes (FIG. **2**):

Each of the storage volumes together with its corresponding file system (**11**) on the storage system (**10**) of the server (**3**) needs to be exclusively allocated and exported to a specific wireless device (**1**) by the console support software (**5**) of the server (**3**).

Data and Storage Volume Management (FIG. **2**)

With the support of the console support software modules (**5**) of the server (**3**), the user of the wireless device (**1**) can via the web-browser **8** of FIG. **2** setup folder/directory structure on the file system of his/her assigned external storage volume (**11**). In addition, the user of the wireless device (**1**) can via the web-browser **8** of FIG. **2** perform all data management operations such as delete, copy, move, rename etc. for file system.

In order to support such data management on the external storage (**10**) allocated to the wireless device (**1** of FIG. **2**) via the web-browser **8** of FIG. **2**, first the console support software modules (**5**) of the server (**3**) must communicate with the web-browser (**8**) of the wireless device (**1**). Therefore, the user from the web-browser (**8**) of the wireless device (**1**) can choose desired data management operations and send operation information to the console support software modules (**5**) of the server (**3**). The mentioned operations include establishing folder/directory, copying, moving, or reaming data file etc. Second, upon receiving the data management operation, the console support software modules (**5**) of the server (**3**) actually processes/executes these requested operations for the assigned file system of an allocated storage volume (**11**) on the server (**3**).

Store Data from Wireless Device into External Storage (FIG. **2**)

To store the data such as digital photo pictures or messages into the file system on the allocated storage (**10**) of the server (**3**), the other software modules (**9**) of the wireless device (**1**) need to send these data to the other service modules (**7**) of the server (**3**) via communication link between them. Upon receiving the data, the other service modules (**7**) of the server (**3**) write these data to the file system of the allocated storage volume (**11**) for the wireless device **1**. The protocol used between these two communication entities could be either IP or non-IP based protocol.

Download Data from a Remote Web Server Site into Allocated Storage Volume

Now, referring to FIG. **3**, If a user of the wireless device (**1**) wants to download a data from a remote web server (**15**) into the allocated file system (**11**) of the allocated storage volume (**11**) on the server (**3**), the following steps are required:

1) The user of the wireless device (**1**) via a web-browser (**8**) access to a remote downloading web server site (**15**) to obtain information for the downloading via path (a) of FIG. **3**. For example, the user access to a web-page which contains the data name for the downloading.

2) The other software modules (**9**) of the wireless device (**1**) obtain the downloading information, which becomes available in the cached web-pages on the wireless device (**1**) after the web-browser (**8**) access to the web site (**15**).

3) The other software modules (**9**) of the wireless device (**1**) send the obtained downloading information to other service modules (**7**) of the storage server (**3**) via path (b) of FIG. **3**.

4) Upon receiving the downloading information from the wireless device (**1**), the other service module (**7**) of the storage server (**3**) sends a web download request to the web-site (**15**) via path (c) of FIG. **3** and receives the downloading data streams from the web server of the web-site (**15**).

5) Upon receiving downloading data streams, the other service modules (**7**) of the storage server (**3**) write the data streams into the allocated file system (**11**) on the server (**3**) for the wireless device (**1**).

Retrieve Data from Allocated Storage for the Wireless Device

1) If a web-browser (**8**) on a wireless device **1** has embedded video or music functionality, a user of the wireless device (**1**) can use the browser to retrieve and play multimedia data file such as video or music stored in the allocated storage volume (**10**) located on the server (**3**).

2) If there is a needs, the other software module (**9**) of the wireless device (**1**) also can retrieve data file from the allocated file system of the allocated storage volume (**11**) on the server (**3**).

Support External Storage for a Large Number of the Wireless Devices

If there is a need to provide each user a 2 GB of storage space, then a 160 GB disk drive can support 80 users. A 4096 GB (4 Tera Bytes) storage system on the server unit can support 2024 user. Each of the server units only can efficiently support a limited size of the storage system. In order to support a large number of the wireless devices, such as for 500,000 wireless devices, a larger number of the servers is required, in this case 250 servers is required. In order to let a larger number of the servers to effectively support the larger number of the wireless devices, an infrastructure like CCDSVM is desirable, which has been described in previous patent applications. With the CCDSVM the control management station can control larger number of storage servers to provide external storage for a huge number of the wireless devices.

The invention claimed is:

1. A server providing storage space to wireless device, the server comprising:

at least one storage device, and

a storage medium comprising program code that, when executed by the server, causes the server to:

allocate, via the at least one storage device, a storage space of a predefined capacity to a wireless device,

create a folder structure residing in the storage space, wherein the storage space being remotely located with respect to the wireless device;

establish a link for the wireless device access to the folder structure in the storage space; and

couple with the wireless device for allowing a user thereof access to the folder structure in the storage space for storing a data object therein or retrieving a data object therefrom, the storing of a data object including to download a file from a remote web server directly into the storage space.

2. The server as recited in claim **1**, wherein the access to the folder structure comprises: creating, from the wireless device, a folder in the folder structure.

3. The server as recited in claim **1**, wherein the data object further is a message, or a multimedia data of video, digital music or photo picture.

4. The server as recited in claim **2**, wherein the access to the folder structure comprises: storing, from the wireless device, a data object into the folder in the folder structure; or deleting or moving or copying or renaming, from the wireless device, a file or a folder in the folder structure.

5. The server as recited in claim **1**, wherein the folder structure is created with multiple folders on a file system configured in the storage space accessible to the wireless devices.

6. The server as recited in claim **4**, wherein the user of the wireless device is allowed via a web browser executed thereof to perform operation to the file or folder in the folder structure.

7. The server as recited in claim **1**, wherein the link is wireless.

8. The server as recited in claim **1**, wherein the downloading of a file comprises:

obtaining download information for the file from the remote web server;

transmitting the download information cached in the wireless device to the server; and

causing the server in accordance with the downloading information to download the file from the remote web server directly into the storage space.

9. The server as recited in claim **7**, further comprising: facilitating a console for an administrator of a service provider to partition the at least one storage device for allocating the storage space to the wireless device.

10. A server comprising:

a plurality of storage devices;

wherein at least a first one of the storage devices is configured with a storage space of a predefined capacity allocated to a wireless device and allow the wireless device via a wireless link access to the storage space, said access to the storage space including to store a data object therein retrieve a data object therefrom,

wherein the storage space is remotely located with respect to the wireless device, and

wherein the server is configured to couple with the wireless device for allowing a user on the wireless device to download a file from a remote web server directly into the allocated storage space.

11. The server as recited in the claim **10**, wherein said downloading of a file comprises:

obtaining download information for the file from the remote web server;

transmitting the download information cached in the wireless device to the server; and

causing the server in accordance with the downloading information to download the file from the remote web server directly into the storage space allocated to the wireless device.

12. The server as recited in claim **10**, wherein the server coupled with the wireless device for allowing the user thereof performing an operation for a data object in the storage space, including to store a file or create a folder therein, or delete or move or copy or rename a file or a folder stored therein.

13. The server as recited in claim **10**, wherein the data object further is a file.

14. The system as recited in claim **10**, further comprising: at least a second one of the storage devices is configured with a storage space of a predefined capacity allocated to a wireless device and allow the wireless device via a wireless link access to the storage space.

15. The server as recited in claim **10**, wherein the data object further is a message, or a multimedia data of video or digital music or photo picture.

16. A non-transitory computer-readable medium, residing in a server, comprising:

program instructions that, when executed by the server, cause the server to:

allocate a storage space of a predefined capacity for each of a plurality of wireless devices;

establish a wireless link for the each of the wireless devices access to the storage space allocated; and

couple with the each of the wireless devices for allowing a user thereof access to the storage space for storing a data object therein or retrieving a data object therefrom, the storing of a data object including to download a file from a remote web server directly into the storage space.

17. The computer-readable medium of claim **16**, wherein said access to the storage space comprises: creating, from the each of the wireless devices, a folder in the storage space.

18. The computer-readable medium of claim **16**, wherein the data object further is a message or a multimedia data of video, music, or picture.

19. The computer-readable medium of claim **17**, wherein the access to the storage space comprises: storing, from the each of the wireless devices, a data object into the folder in the storage space; or deleting or moving or copying or renaming, from the each of the wireless device, a file or folder in the storage space.

20. The computer-readable medium of claim **16**, wherein said downloading of a file further comprises:

obtain, by the each of the wireless devices, the downloading information for the file from the remote web server;

transmitting the downloading information cached in the each of the wireless devices to the server; and

causing the server in accordance with the downloading information to download the file directly from the remote web server into the storage space allocated to the each of the wireless devices.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 8,868,690 B2
APPLICATION NO. : 14/079831
DATED : October 21, 2014
INVENTOR(S) : Sheng Tai Ted Tsao

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

A) IN THE SPECIFICATION:

In col. 3, line 25, please delete the “implemented”;
In col. 3, line 27, please replace “server (3)” with ---server (3) or---;

B) IN THE CLAIMS:

In col. 6, line 51, Claim 10, please replace the “retrieve” with ---or retrieve---.

Signed and Sealed this
Twenty-seventh Day of January, 2015

Michelle K. Lee
Deputy Director of the United States Patent and Trademark Office

EXHIBIT F

US009219780B1

(12) **United States Patent**
Tsao

(10) **Patent No.: US 9,219,780 B1**
(45) **Date of Patent: *Dec. 22, 2015**

(54) **METHOD AND SYSTEM FOR WIRELESS DEVICE ACCESS TO EXTERNAL STORAGE**

(71) Applicant: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(72) Inventor: **Sheng Tai Tsao**, Fremont, CA (US)

(73) Assignee: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **14/623,476**

(22) Filed: **Feb. 16, 2015**

Related U.S. Application Data

(63) Continuation of application No. 14/150,106, filed on Jan. 8, 2014, now Pat. No. 9,098,526, which is a continuation of application No. 14/079,831, filed on Nov. 14, 2013, now Pat. No. 8,868,690, which is a continuation of application No. 10/726,897, filed on Dec. 4, 2003, now Pat. No. 8,606,880.

(51) **Int. Cl.**

G06F 15/16	(2006.01)
H04L 29/08	(2006.01)
H04W 76/02	(2009.01)
G06F 17/30	(2006.01)

(52) **U.S. Cl.**
CPC ***H04L 67/10*** (2013.01); ***G06F 17/30197*** (2013.01); ***H04L 67/02*** (2013.01); ***H04W 76/023*** (2013.01)

(58) **Field of Classification Search**
USPC 709/219, 203, 226, 220, 200; 455/412.1, 899
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

8,606,880 B2 *	12/2013	Tsao	G06F 17/30067	455/412.1
2001/0052073 A1 *	12/2001	Kern	G06F 21/10	713/161
2002/0104080 A1 *	8/2002	Woodard	G06F 9/44505	717/176
2002/0133597 A1 *	9/2002	Jhingan	H04L 67/30	709/228
2003/0072031 A1 *	4/2003	Kuwata	H04N 1/00151	358/1.15
2004/0033821 A1 *	2/2004	Slesak	H04N 5/765	455/575.1
2004/0186750 A1 *	9/2004	Surbey	G06Q 40/08	705/4

* cited by examiner

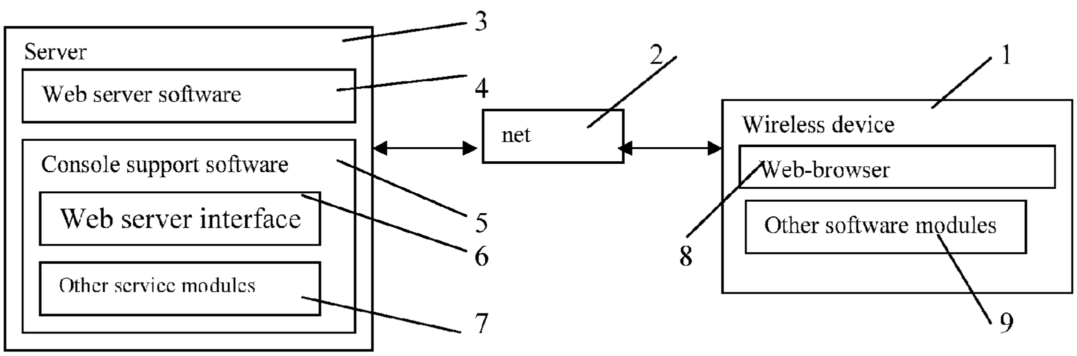
Primary Examiner — El Hadji Sall

(57) **ABSTRACT**

Traditionally, wireless device, such as cell phone or personal data assistant device (PDA), has relatively smaller storage capacity. Therefore, it is quite often that a user of the wireless device has difficulty to find more storage space for storing ever increased personal data, such as storing multiple Gig bytes of multimedia data including digital video, music, or photo picture etc.. Instant application disclosed a system and method for a storage system providing storage service to the wireless device for the wireless device remotely storing personal data into an external storage space allocated exclusively to a user of the wireless device by the storage system.

20 Claims, 4 Drawing Sheets

Wireless devices supports in a simple environment



Wireless devices supports in a simple environment

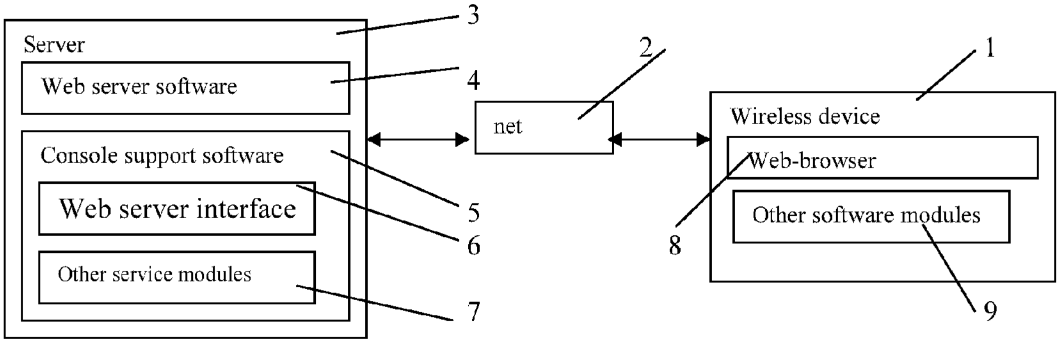


Fig. 1

Wireless devices access external storage through web browser

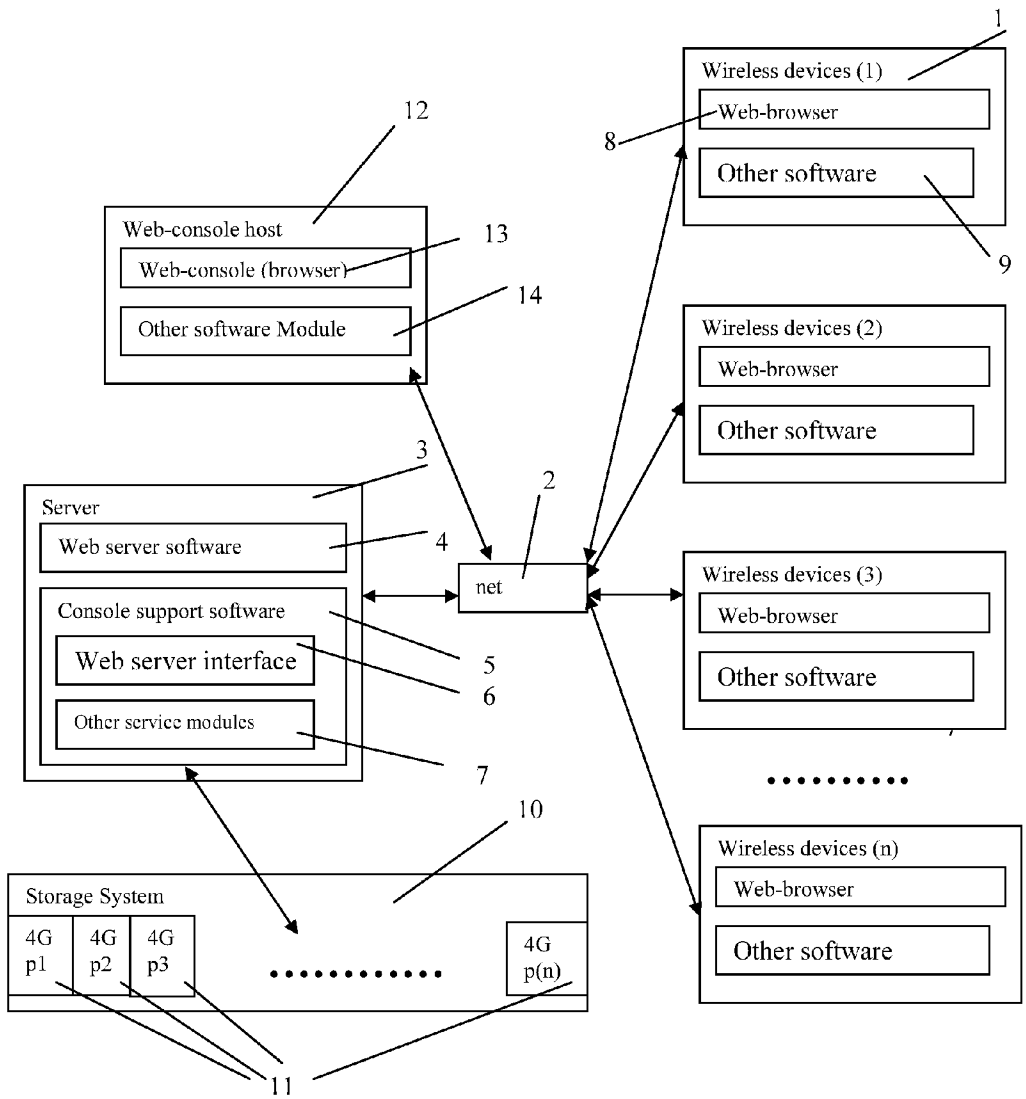


Fig. 2

Wireless out-band download

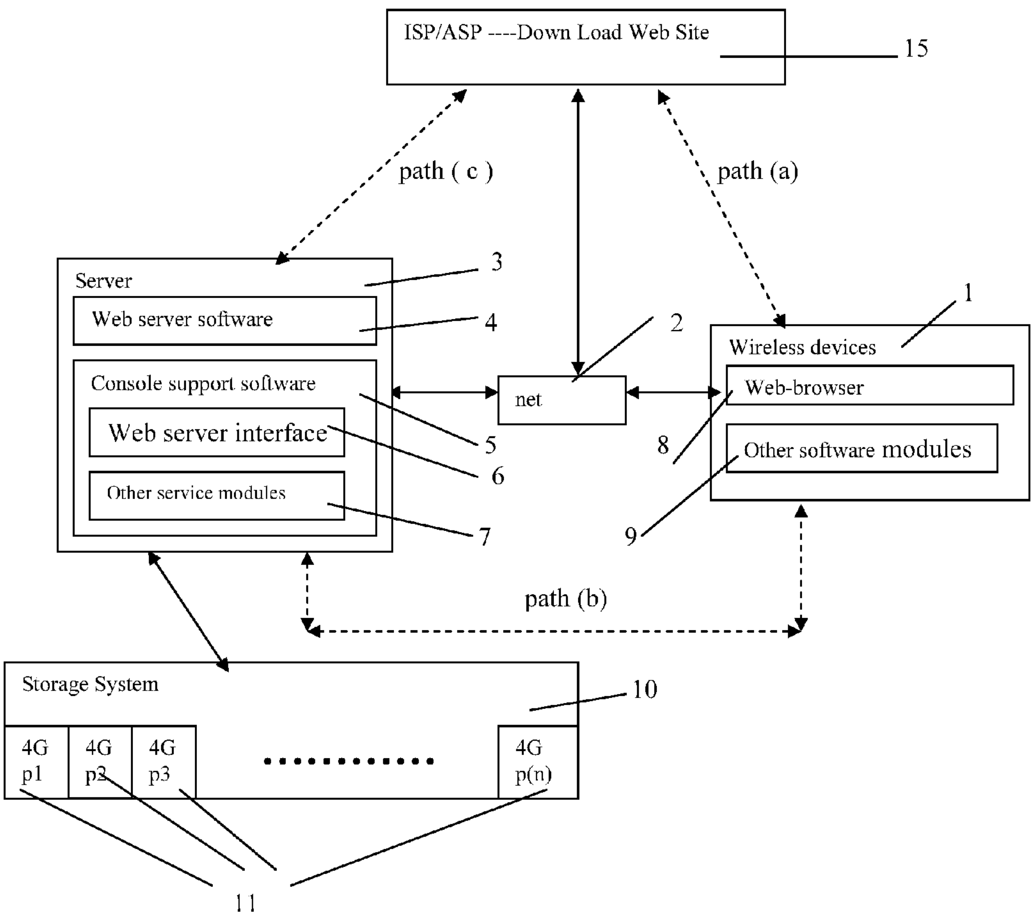


Fig. 3

The CCDSVM Support External Device for Huge Number of Wireless Device

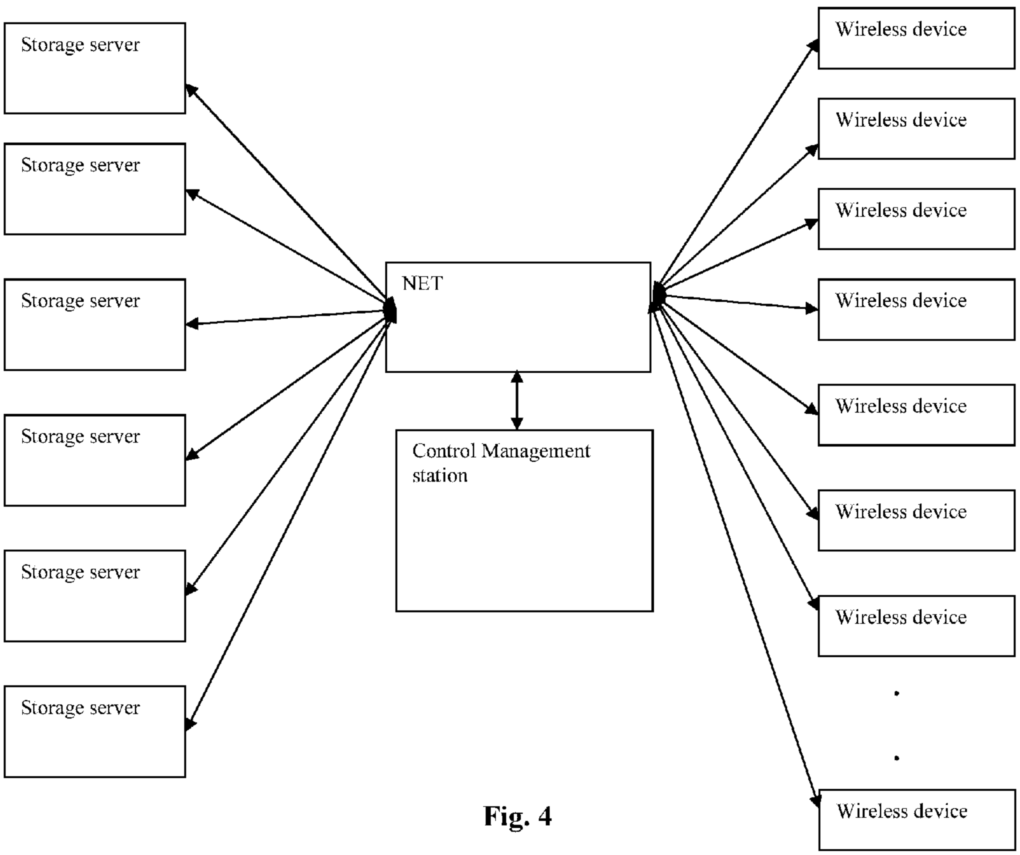


Fig. 4

METHOD AND SYSTEM FOR WIRELESS DEVICE ACCESS TO EXTERNAL STORAGE

CROSS REFERENCE TO PRIOR APPLICATION

This invention is a continuation application of the U.S. patent application Ser. No. 10/726,897 filed on Dec. 4, 2003 and now a U.S. Pat. No. 8,606,880. The application Ser. No. 10/726,897 had referenced a prior application No. 60/401,238 of "Concurrent Web Based Multi-task Support for Control Management System" filed on Aug. 6, 2002 and converted to U.S. patent application Ser. No. 10/713,904 filed on Jul. 22, 2003 and now is an U.S. Pat. No. 7,418,702, and had also referenced an prior application No. 60/402,626 of "IP Based Distributed Virtual SAN" filed on Aug. 12, 2002 and converted to application Ser. No. 10/713,905 filed on Jul. 22, 2003 and now is an U.S. Pat. No. 7,379,990. All mentioned prior applications are herein incorporated by reference in their entirety for all purpose.

FIELD OF THE INVENTION

This invention focuses on a wireless device accessing and using external storage space provided by a server.

BACKGROUND INFORMATION

Storage system always is a critical part of a computing system regardless of the computing system is a server, a laptop or desktop computer, or a wireless device as cell phone or personal data assistant device ("PDA"). The storage system can be categorized as internal storage or external storage system.

The internal storages of a computing system include those storage media such as hard disk drives, memory sticks, and memory etc. that are internally connected in the computing system directly through system bus or a few inches of cable, Therefore, the storage media actually are internal components of the computing system in a same enclosure.

The external storages of a computing system are those storage media that are not the internal components of the computing system in a same enclosure. Therefore, the storage media of the computing system have to be accessed through longer cable, such as through Ethernet controller with longer cable for IP based storage, Fiber channel controller with longer cable for fiber channel storage, or wireless communication medium, etc.. The storage media of the external storage could be magnetic hard disk drives, solid state disk, optical storage drives, memory card, etc. and could be in any form such as Raid which usually consists of a group of hard disk drives.

To effectively use a storage system, the storage devices of the storage system usually need to be partitioned into storage volumes. After the partition, each of the volumes can be used for establishing a file system on top of it. To simplify the discussion, herein, the term of the storage volume and its corresponding file system, and the storage partition are often used without differentiation.

To satisfy the needs for external storage for a larger number of wireless devices, a central controlled distributed scalable virtual machine "CCDSVM" can be deployed. The CCDSVM allows a control management system to control a group of computing systems for providing distributed services, including storage service, to client devices over the Internet, Intranet, and LAN environment.

As a matter of the fact, today major Internet service provider (ISP) and application service provider (ASP) are all in business of providing various type of storage services to their clients.

BRIEF DESCRIPTION OF FIGURES

FIG. **1** illustrates an embodiment of present invention. The FIG. **1** is the same as the FIG. **1** of a prior application of the "Concurrent Web Based Multi-task Support for Control Management System" with an exception of a console host being replaced by a wireless device.

FIG. **2** is the same as the FIG. **1** of the above except that it illustrates an embodiment of a more detailed storage system controlled by a server. In addition, multiple wireless devices are presented for access to the storage system.

FIG. **3** shows a scheme of a wireless device downloading contents from an ISP/ASP or from other web sites to an external storage allocated for access by the wireless device.

FIG. **4** is the same figure as the FIG. **1** of the prior application of "IP Based Distributed Virtual SAN" with an exception that each IP storage server provides file system as external storage for each of the wireless devices instead of providing IP based virtual SAN service. Also, each host in the FIG. **1** now actually is a wireless device in the FIG. **4**.

Unless specified, the programming languages and the protocols used by the software modules, and the computing systems used of present invention are assumed to be the same as described in the prior patent applications.

In addition, in the drawing, like elements are designated by like reference numbers. Further, when a list of identical elements is present, only one element will be given the reference number.

BRIEF DESCRIPTION OF THE INVENTION

Today, users commonly face a problem of lack of storage capacity configured on their wireless devices such as cell phone or PDA, which are usually limited to 256 MB for the PDA and much less for the cell phone. To effectively solve this problem and let users possess multiple gigabytes (GB) of storage for their wireless devices as well as allowing the users to use the GB storage for their multimedia applications, the storage spaces provided by a server can be used as the external storage of the wireless devices. This technology has been briefly introduced in the prior patent applications.

Now let us examine how can the external storage actually be used by the wireless devices. First, let each server unit (e.g. the server **3** of the FIG. **2**) partition its storage system into volumes, such that each of the volumes will have multiple GB in size. Therefore, a user of each of the wireless devices can be exclusively assigned for access to a specific storage volume in the server unit. For example, if we need to provide each user a 4 GB storage space, then a 160 GB disk drive can support 40 users. Therefore, a 4096 GB storage system of the server unit can support a total of 1024 users. Further, any data on a wireless device of the user can be transmitted to the assigned storage volume in the server unit. In addition, the user of the wireless device also can download multimedia data from an ISP or ASP to the assigned storage volume in the server unit through out-band approach shown in FIG. **3**. Finally, in one embodiment, the user can use a web-browser, which has a functionality of invoking embedded video or music, to enjoy his/her stored multimedia contents.

These and other futures, aspects and advantages of the present invention will become understood with reference to the following description, appended claims, and accompanying figures.

DETAILED DESCRIPTION OF THE DRAWINGS

Referring now to the FIG. **1**, it demonstrates an example of a network infrastructure which includes a wireless device and a server. In the FIG. **1**, Net (**2**) represents a communication link, which may be combined with wireless and wired connection media and guarantee that the communication packets can be sent/received between the wireless device and the server. It is also assumed that the net (**2**) representing the network infrastructure is built up in such way that a user from a web-browser of a wireless device can access and browse any web-site on the Internet, and Intranet.

In the FIG. **1**, the console support software (**5**) of the server (**3**) can be configured to support web-based multi-tasking while a user on a wireless device (**1**) using a web browser (**8**). Further, the user of the wireless device is facilitated to perform creating structured layered file directories or folders, and perform data management operations, such as delete, move, copy, rename for data files or folders or directories etc. on an assigned storage volume controlled by the server (**3**). In addition, the other software modules (**9**) of the wireless device (**1**) is also configured capable of to send data to or receive data from the other service modules (**7**) of the server (**3**) via communication link (**2**) through a suitable IP or non-IP based protocol. The data being sent or received cold be a digital photo picture, a message without limits.

Also, the console supporting software (**5**) of the server (**3**) and the other software modules (**9**) of the wireless device (**1**) can be implemented with any suitable languages such as C, C++, Java without limits.

Besides, the web-browser (**8**) of the wireless device (**1**) can be any suitable software, which, is capable to communication with web server software (**4**) on the server (**3**) or with any other web server through the HTTP protocol.

On the other hand, FIG. **2** has demonstrated the detailed storage system (**10**) of a server **3**, where its storage volumes can be allocated to multiple wireless devices as followings: First, the storage system (**10**) of the server (**3**) can be partitioned into multiple storage volumes (**11**), for example, by administration staff through a web-console (**13**) of a console host (**12**).

Second, the storage system (**10**) of the server (**3**) can be partitioned in such way that a user of each of the wireless devices can be assigned with a storage volume having a desired size, so that the server **3** can support maximum numbers of the wireless devices.

In addition, the storage connection media could be any kind, such as SCSI cable, IP cable, Fiber cable etc. or could be wireless communication media. The storage system itself could be various types.

Finally, the storage system (**10**) can be accessed by each of the wireless devices through IP or non-IP based network and protocols.

The FIG. **3** has demonstrated that a user from a web-browser (**8**) on a wireless device (**1**) can download data from a known web-site (**15**) to his/her assigned external storage (**10**) on the server (**3**). The dash-lined path (a) represents a communication link between the wireless device (**1**) and a remote web-site (**15**) that provides downloading contents. The dash-lined path (b) represents a communication link between the wireless devices (**1**) and the storage server (**3**). The dash-lined path (c) represents a communication link between the server **3** and the remote web-server (**15**).

THE DETAILED DESCRIPTION OF THE INVENTION

The Use of the External Storage by the Wireless Device:

The FIG. **2** shows a simplified diagram of the wireless devices (**1**) using the external storage system (**10**) of the server (**3**) for effectively resolving the storage limitation problem for the wireless devices (**1**).

Partition Storage Volumes (FIG. **2**):

With this invention, the entire storage (**10**) on the server (**3**) needs to be partitioned into suitable size of volumes (**11**) such as 4 GB for each volume. This will allow the server **3** to serve maximum number of the wireless devices (**1**). With the web console support software (**5**) of the server (**3**), tasks of partitioning the storage system (**10**) can be done through a web-console (**13**) on a console host (**12**) by an administrative staff.

In order to support storage partition, first the console support software (**5**) of the server (**3**) must send storage information of the server (**3**) to the web-console (**13**) of the console host (**12**). The storage information includes each storage device's name and total size etc. without limits. Second, based on the received storage information the administration staff on the console host (**12**), for example, can use a web-console (**13**) to partition each storage device and send the storage partition information to the console support software (**5**) of the server (**3**). The storage partition information includes the number of the partitions (volumes) and the size of each partition (volume). Third, upon receiving the storage partition information from the web-console (**13**) of the console host (**12**), the console support software (**5**) of the server (**3**) performs the actual storage partition by dividing the entire storage into multiple small volumes. Finally, for each small storage volume, a corresponding file system could be built up.

Assign Storage Volumes (FIG. **2**):

Each of the storage volumes (**11**) together with its corresponding file system on the storage system (**10**) of the server (**3**) needs to be exclusively assigned to a user of a specific wireless device (**1**) by the console support software (**5**) of the server (**3**).

Data and Storage Volume Management (FIG. **2**)

With the support of the console support software modules (**5**) of the server (**3**) by following similar steps of partitioning storage, a user of the wireless device (**1**) can use a web-browser **8** illustrated in FIG. **2** to setup folder/directory structure on the file system of his/her an assigned external storage volume (**11**). In addition, the user of the wireless device (**1**) can use the web-browser **8** performing all data management operations such as delete, copy, move, rename for file or folder on that file system.

In order to support such data management over the external storage (**10**) assigned to the user of the wireless device (**1**) by using the web-browser **8**, first the console support software modules (**5**) of the server (**3**) must communicate with the web-browser (**8**) of the wireless device (**1**) to present the assigned storage to a user as described before. Therefore, the user from the web-browser (**8**) of the wireless device (**1**) can choose a desired data management operation and send information of the operation to the console support software modules (**5**) of the server (**3**). The mentioned operation includes establishing folder/directory, copying, moving, or reaming data file etc. for the folder directory. Second, upon receiving the data management operation, the console support software modules (**5**) of the server (**3**) actually performs these requested operations on the assigned file system of an assigned external storage volume (**11**) on the server (**3**).

Store Data from Wireless Device into External Storage (FIG. **2**):

To store the data such as digital photo pictures or messages into the file system on the assigned storage volume (**11**) in the server (**3**), the other software modules (**9**) of the wireless device (**1**) need to send these data to the other service modules (**7**) of the server (**3**) via communication link between them. Upon receiving the data, the other service modules (**7**) of the server (**3**) write these data to the file system of the assigned storage volume (**11**) on server (**3**) for the wireless device (**1**). The protocol used between these two communication entities could be either IP or non-IP based protocol.

Download Data from a Remote Web Server Site into Allocated Storage Volume:

Now, referring to the FIG. **3**, If a user of the wireless device (**1**) wants to download a data from a remote web server (**15**) into the file system on the assigned storage volume (**11**) in the external storage system (**10**) on the server (**3**), the following steps are required:

1) The user of the wireless device (**1**) via a web-browser (**8**) access to a remote web server site (**15**) to obtain information of the data for the downloading via the path (a) of the FIG. **3**. For example, the user access to a web-page which contains the data name for the downloading and also contains IP address of the remote web site.

2) The other software modules (**9**) of the wireless device (**1**) obtain the downloading information for the data, which becomes available in the cached web-pages on the wireless device (**1**) after the web-browser (**8**) access to the web site (**15**).

3) The other software modules (**9**) of the wireless device (**1**) send the obtained downloading information to other service modules (**7**) of the storage server (**3**) via the path (b) of the FIG. **3**.

4) Upon receiving the downloading information from the wireless device (**1**), the other service module (**7**) of the storage server (**3**) sends a web download request to the web-site (**15**) via the path (c) of the FIG. **3** based on download information obtained and then receives the downloading data from the web server of the web-site (**15**).

5) Upon receiving downloading data stream, the other service modules (**7**) of the storage server (**3**) write the data into the file system on the assigned storage volume (**11**) on the server (**3**) for the wireless device (**1**).

Retrieve Data from Assigned Storage Volume for the User of the Wireless Device:

1) If a web-browser (**8**) on a wireless device **1** has embedded video or music functionality, a user of the wireless device (**1**) can use the browser to retrieve and play multimedia data file such as video or music stored in the assigned storage volume (**11**) located on the server (**3**).

2) In another embodiment, in respect to the user's needs, the other software module (**9**) of the wireless device (**1**) also can retrieve data file from the file system of the assigned storage volume (**11**) on the server (**3**).

Support External Storage for a Large Number of the Wireless Devices:

If there is a need to provide each user a 2 GB of storage space, then a 160 GB disk drive can support 80 users. A 4096 GB (4 Tera Bytes) storage system on the server unit can support 2024 user. Each of the server units only can efficiently support a limited size of the storage system. In order to support a large number of the wireless devices, such as for 500,000 wireless devices, a larger number of the servers is required, in this case 250 servers is required. In order to let a larger number of the servers to effectively support the larger number of the wireless devices, an infrastructure like the CCDSVM is desirable, which has been described in prior patent applications. With the CCDSVM the control management system can control larger number of storage servers to provide external storage for a huge number of the wireless devices.

The invention claimed is:

1. A wireless device access to a remote storage space, the wireless device comprising:

at least one cache storage, and one non-transitory computer-readable medium comprising program instructions which, being executed by the wireless device, cause the wireless device remotely access to the storage space, the program instructions include:

program instructions for establishing a wireless link for remotely access to the storage space, the storage space allocated exclusively by a storage server to a user of the wireless device;

program instructions for presenting the storage space to the user on the wireless device through communication with the storage server; and

program instructions for coupling with the storage server across the wireless link to carry out a requested operation for remotely access to the storage space in response to the user from the wireless device performed the operation,

wherein the program instructions for carrying out operation for the access to the storage space comprises program instructions for storing data therein or retrieving data therefrom,

the program instructions for storing data including program instructions for downloading a file from a remote server across the Internet into the storage space through utilizing download information for the file, including name of the file and internet protocol ("IP") address of the remote server, cached in the cache storage in response to the user from the wireless device performed the operation for the downloading.

2. The wireless device of claim **1**, wherein the data being stored into or retrieved from the storage space comprises one of a message, a digital video, a digital music, a digital picture.

3. The wireless device of claim **1**, wherein said program instructions for downloading a file from a remote server comprises program instructions for:

obtaining, by the wireless device, downloading information for the file;

transmitting the downloading information cached in the wireless device to the storage server; and

causing the storage server in accordance with the downloading information to download the file into the storage space.

4. The wireless device of claim **1**, wherein said operation for access to the remote storage space comprises:

creating, from the wireless device, a folder structure in the storage space.

5. The wireless device of claim **1**, wherein said operation for access to the remote storage space comprises:

deleting or moving or rename or copying, from the wireless device, a folder in the remote storage space.

6. The wireless device of claim **1**, wherein said operation for access to the remote storage space comprises:

deleting or moving or rename or copying, from the wireless device, a file in the remote storage space.

7. The wireless device of claim **1**, wherein the wireless device is one of a cell phone or a personal data assistant and management device ("PDA").

8. The wireless device of claim **1**, wherein the wireless device further executes a web browser for the user access to the remote storage space in addition for access to the Internet.

9. A server comprising:

a pool of a plurality of storage spaces, and non-transitory computer-readable storage medium comprising program instructions which, being executed by the server, causes the server delivering storage service, the program instructions include:

program instructions for allocating exclusively, via the storage pool, a first one of the storage spaces to a user of a first wireless device;

program instructions for establishing a communication link for the first wireless device remotely access to the first one of the storage spaces;

program instructions for sending information of the first one of the storage spaces to the first wireless device for presenting the first one of the storage spaces to the user on the wireless device; and

program instructions for updating the first one of the storage spaces in response to the user from the first wireless device performed an operation for said remotely access to the first one of the storage spaces,

wherein said access to the first one of the storage spaces comprises storing data therein or retrieving data therefrom,

the storing of a data object including to download a file from a remote server into the first one of the storage spaces through utilizing download information for the file, including name of the file and internet protocol ("IP") address of the remote server, cached in a cache storage of the first wireless device in response to the user from the first wireless device performed the operation for the downloading.

10. The server of claim **9**, wherein said downloading a file from the remote server comprises:

obtaining, by the first wireless device, the downloading information for the file from the remote server;

transmitting the downloading information cached in the first wireless device to the server; and

causing the server in accordance with the downloading information to download the file into the first one of the storage spaces.

11. The server of claim **9**, wherein said operation for access to the first one of the storage spaces comprises:

creating, from the first wireless device, a folder in the first one of the storage spaces.

12. The server of claim **9**, wherein the server further allocates exclusively, via the storage pool, a second one of the storage spaces to a user of a second wireless device for exclusive access.

13. The server of claim **9**, wherein the first wireless device further executes a web browser on the first wireless device for the user access to the first one of the storage spaces, in addition for access to the Internet.

14. The server of claim **9**, wherein said operation for access to the first one of the storage spaces comprises: deleting or moving or rename or copying, from the first wireless device, a file or a folder in the first one of the storage spaces.

15. The server of claim **9**, wherein the data object being stored into or retrieved from the first storage space comprises one of a message, a digital video, a digital music, a digital picture.

16. A system comprising:

at least one storage server and one wireless device;

wherein the storage server comprises a plurality of storage spaces, a first one of which being allocated to a user of the wireless device for exclusive access, and causes presenting the first one of the storage spaces to the user on the wireless device, and

updates the first one of the storage spaces in response to the user from the wireless device performed an operation for remotely access to the first one of the storage spaces; and

wherein the wireless device couples with the storage server across a wireless link to carry out a requested operation for remotely access to the first one of the storage spaces in response to the user from the wireless device performed the operation for the access,

wherein the operation for remotely access to the first one of the storage spaces comprises storing data therein or retrieving data therefrom,

the storing of said data including to download a file from a remote server into the first one of the storage spaces through utilizing download information for the file cached in a cache storage in the wireless device in response to the user from the wireless device performed the operation for the downloading.

17. The system of claim **16**, wherein said downloading a file from a remote server comprises:

obtaining, by the wireless device, downloading information for the file from the remote server;

transmitting the downloading information cached in the wireless device to the storage server; and

causing the storage server in accordance with the downloading information to download the file into the first one of the storage spaces.

18. The system of claim **16**, wherein the wireless device is one of a cell phone or a personal data assistant and management device ("PDA").

19. The system of claim **16**, wherein said download information for the file includes name of the file and internet protocol ("IP") address of the remote server stored in the cache storage.

20. The system of claim **16**, further comprising: the storage server allocating a second one of the storage spaces to a user of a second wireless device for the user from the second wireless device remotely access to the second one of the storage spaces.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 9,219,780 B1
APPLICATION NO. : 14/623476
DATED : December 22, 2015
INVENTOR(S) : Sheng Tai Ted Tsao

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

In the claims

1) In col. 6, line 29 & line 41, and
col. 7, line 24 & line 35, and
col. 8, line 25 & line 34, please replace “performed” with ---performing---;

2) In col. 7, line 29, please replace “a data” with ---the data---;

3) In col. 7, line 29 and col. 8, line 5, please remove “object”.

Signed and Sealed this
Eighth Day of March, 2016

Michelle K. Lee

Michelle K. Lee
Director of the United States Patent and Trademark Office

EXHIBIT G

US009239686B2

(12) **United States Patent**
Tsao

(10) **Patent No.: US 9,239,686 B2**
(45) **Date of Patent: *Jan. 19, 2016**

(54) **METHOD AND APPARATUS FOR WIRELESS DEVICES ACCESS TO EXTERNAL STORAGE**

(71) Applicant: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(72) Inventor: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(73) Assignee: **Sheng Tai (Ted) Tsao**, Fremont, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 179 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **14/036,744**

(22) Filed: **Sep. 25, 2013**

(65) **Prior Publication Data**

US 2015/0089181 A1 Mar. 26, 2015

Related U.S. Application Data

(63) Continuation of application No. 10/726,897, filed on Dec. 4, 2003, now Pat. No. 8,606,880.

(51) **Int. Cl.**
G06F 13/00 (2006.01)
G06F 3/06 (2006.01)

(52) **U.S. Cl.**
CPC ***G06F 3/0631*** (2013.01); ***G06F 3/0604*** (2013.01); ***G06F 3/067*** (2013.01); *G06F 2003/0697* (2013.01)

(58) **Field of Classification Search**
CPC G06F 3/067; G06F 3/0604; G06F 3/0631; G06F 2003/0697
USPC 711/111, 114, 147, 171; 709/219, 226; 707/795, 802
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

6,754,710	B1 *	6/2004	McAlear	709/227
6,839,743	B2 *	1/2005	Shim	G06F 17/3089 707/E17.116
6,968,334	B2 *	11/2005	Salmenkaita et al.	707/10
7,286,651	B1 *	10/2007	Packingham et al.	379/88.14
7,707,065	B1 *	4/2010	Kangas	705/26.1
8,059,818	B2 *	11/2011	Asokan et al.	380/277
8,229,856	B1 *	7/2012	Reddick	G06Q 30/04 705/50
2002/0133597	A1 *	9/2002	Jhingan et al.	709/228
2002/0173295	A1 *	11/2002	Nykanen et al.	455/414
2003/0072031	A1 *	4/2003	Kuwata et al.	358/1.15
2004/0176958	A1 *	9/2004	Salmenkaita et al.	704/275
2004/0186750	A1 *	9/2004	Surbey et al.	705/4
2005/0193199	A1 *	9/2005	Asokan et al.	713/168
2006/0036610	A1 *	2/2006	Wang	707/10
2006/0168123	A1 *	7/2006	Krstulich	H04L 63/083 709/219
2008/0059519	A1 *	3/2008	Grifftih	H04L 67/06
2008/0208870	A1 *	8/2008	Tsang	G06F 17/30165
2010/0211651	A1 *	8/2010	Guedalia et al.	709/217
2010/0325709	A1 *	12/2010	Kawase	H04N 7/16 726/7
2012/0124370	A1 *	5/2012	Choi et al.	713/159
2014/0208399	A1 *	7/2014	Ponzio, Jr.	726/4

* cited by examiner

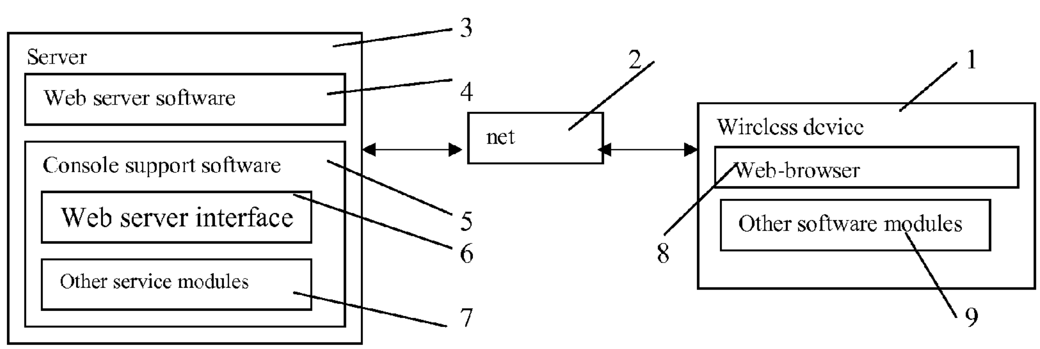
Primary Examiner — Reba I Elmore

(57) **ABSTRACT**

Traditionally, wireless device, such as cell phone or personal data assistant device (PDA), has relatively smaller storage capacity. Therefore, it is quite often that a user of the wireless device has difficulty to find more storage space for storing ever increased personal data, such as storing multiple Gig bytes of multimedia data including digital video, music, or photo picture etc. Instant application disclosed a system and method for a storage system providing storage service to the wireless device for the wireless device remotely storing personal data into an external storage space allocated exclusively to a user of the wireless device by the storage system.

20 Claims, 4 Drawing Sheets

Wireless devices supports in a simple environment



Wireless devices supports in a simple environment

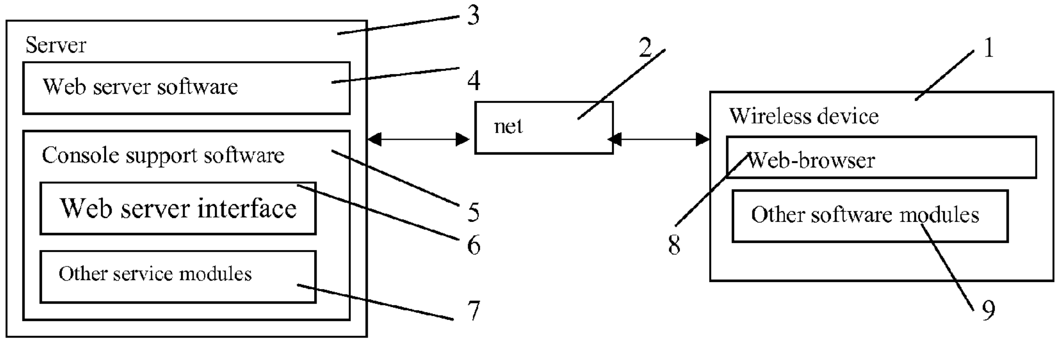


Fig. 1

Wireless devices access external storage through web browser

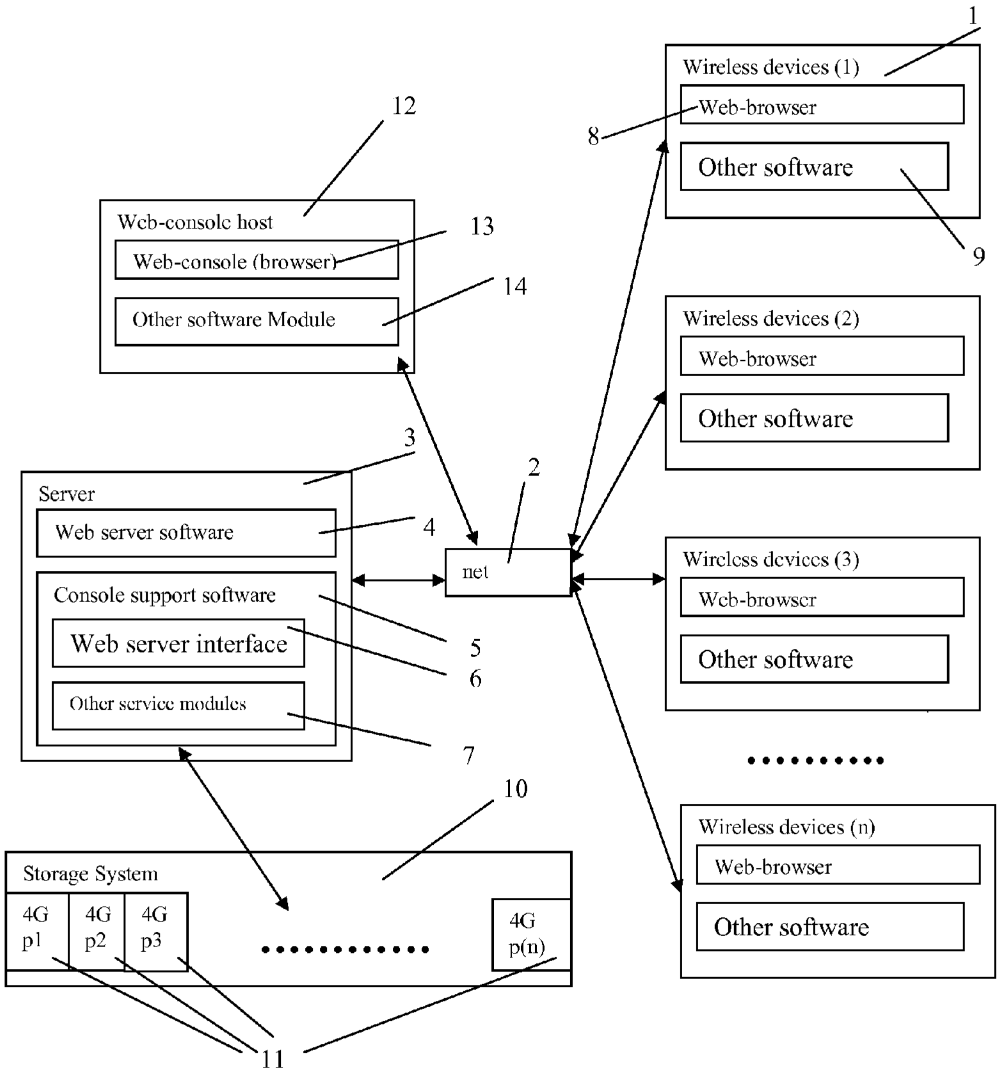


Fig. 2

Wireless out-band download

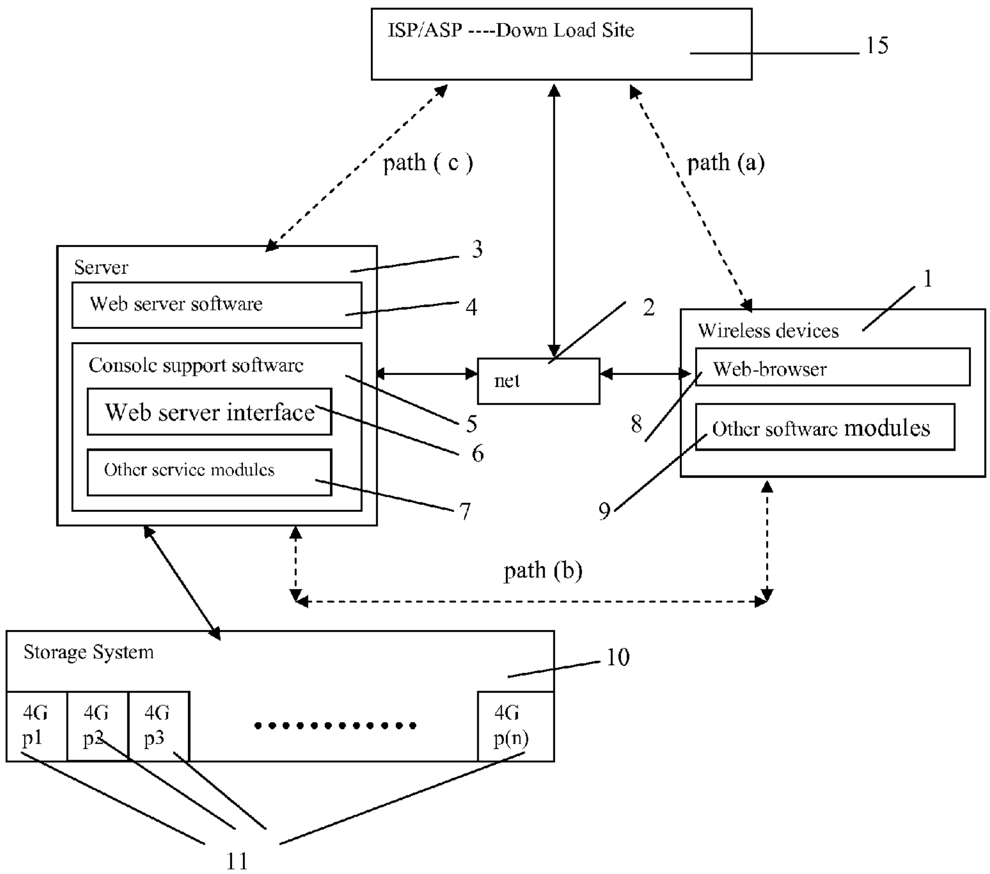


Fig. 3

The CCDSVM Support External Device for Huge Number of Wireless Device

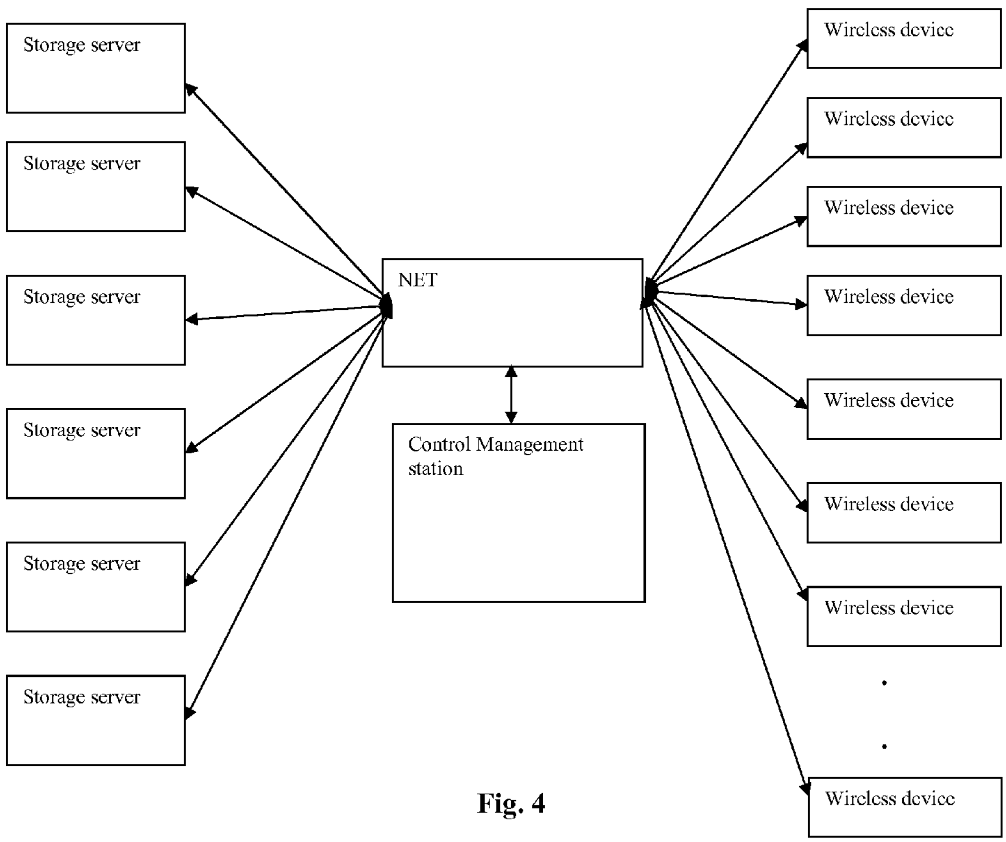


Fig. 4

METHOD AND APPARATUS FOR WIRELESS DEVICES ACCESS TO EXTERNAL STORAGE

CROSS REFERENCE TO PRIOR APPLICATION

This invention is a continuation application of the U.S. patent application Ser. No. 10/726,897 filed on Dec. 4, 2003 and now a U.S. Pat. No. 8,606,880. The application Ser. No. 10/726,897 had referenced to a prior application No. 60/401,238 of "Concurrent Web Based Multi-task Support for Control Management System" filed on Aug. 6, 2002 and converted to U.S. patent application Ser. No. 10/713,904 filed on Jul. 22, 2003 and now is an U.S. Pat. No. 7,418,702, and had also referenced to an prior application No. 60/402,626 of "IP Based Distributed Virtual SAN" filed on Aug. 12, 2002 and converted to U.S. patent application Ser. No. 10/713,905 filed on Jul. 22, 2003 and now is an U.S. Pat. No. 7,379,990. All mentioned prior applications are herein incorporated by reference in their entirety for all purpose.

FIELD OF THE INVENTION

This invention focuses on a wireless device accessing and using external storage space provided by a server.

BACKGROUND INFORMATION

Storage system always is a critical part of a computing system regardless of the computing system is a server, a laptop or desktop computer, or a wireless device such as cell phone or personal data assistant device ("PDA"). The storage system can be categorized as internal storage or external storage system.

The internal storages of a computing system include those storage media such as hard disk drives, memory sticks, and memory etc. that are internally connected in the computing system directly through system bus or a few inches of cable, Therefore, the storage media actually are internal components of the computing system in a same enclosure.

The external storage of a computing system are those storage media that are not the internal components of the computing system in a same enclosure. Therefore, the storage media have to be accessed through longer cable, such as through Ethernet controller with longer cable for IP based storage, Fiber channel cable for fiber channel storage, or wireless communication media etc. The storage media of the external storage could be magnetic hard disk drives, solid state disk, optical storage drives, memory card, etc. and could be in any form such as Raid which usually consists of a group of hard disk drives.

To effectively use a storage system, the storage devices of the storage system usually need to be partitioned with storage volumes. After the partition, each of the volumes can be used for establishing a file system on top of it. To simplify the discussion, herein, the term of the storage volume and its corresponding file system, and the storage partition are often used without differentiation.

To satisfy the needs for external storage for a larger number of wireless devices on the Internet, a central controlled distributed scalable virtual machine ("CCDSVM") of present invention can be deployed. The CCDSVM allows a control management system to control a group of computing systems for providing distributed services, including storage service, to client systems over the Internet, Intranet, and LAN environment.

As matter of the fact, today major Internet service provider (ISP) and application service provider (ASP) are all in business of providing various type of storage services to their clients.

BRIEF DESCRIPTION OF FIGURES

FIG. **1** illustrates an embodiment of the instant application. The FIG. **1** is the same as the FIG. **1** of a prior application of "Concurrent Web Based Multi-task Support for Control Management System" with an exception of a console host being replaced by a wireless device.

FIG. **2** is the same as the FIG. **1** of the above except that it illustrates an embodiment of a more detailed storage system controlled by a server. In addition, multiple wireless devices are presented for access to the storage system.

FIG. **3** shows a scheme of a wireless device downloading contents from an ISP/ASP or from other web sites to an external storage allocated for exclusive access by the wireless device.

FIG. **4** is the same figure as the FIG. **1** of a prior application of "IP Based Distributed Virtual SAN" with an exception that each IP storage server provides file system as external storage for each of the wireless devices instead of providing IP based virtual SAN service. Also, each host in the FIG. **1** now actually is a wireless device in the FIG. **4**.

Unless specified, the programming languages and the protocols used by the software modules and the computing systems used of present invention are assumed to be the same as described in the prior patent applications.

In addition, in the drawing, like elements are designated by like reference numbers. Further, when a list of identical elements is present, only one element will be given the reference number.

BRIEF DESCRIPTION OF THE INVENTION

Today, users commonly face a problem of lack of storage capacity configured on their wireless devices such as cell phone or PDA, which are usually limited to 256 MB for the PDA and much less for the cell phone. To effectively solve this problem and let users possess multiple gigabytes (GB) of storage for their wireless devices as well as allowing the users to use the GB storage for their multimedia applications, the storage spaces provided by a server can be used as the external storage for the wireless devices. This technology has been briefly introduced in the prior patent applications.

Now let us examine how can the external storage actually be used by the wireless devices. First, facilitate each server unit (e.g. the server **3** of the FIG. **2**) partition its storage system into volumes, such that each of the volumes will have multiple GB in size. Therefore, a user of each of the wireless devices can be exclusively assigned for access to a specific storage volume in the server unit. For example, if we need to provide each user a 4 GB storage space, then a 160 GB disk drive can support 40 users. Therefore, a 4096 GB storage system of the server unit can support a total of 1024 users. Further, any data on the wireless device of the user can be transmitted to an assigned storage volume in the server unit. In addition, the user of the wireless device also can download multimedia data from an ISP or ASP to the assigned storage volume in the server unit through out-band approach shown in FIG. **3**. Finally, in one embodiment, the user can use a web-browser, which has a functionality of invoking embedded video or music, to enjoy his/her stored multimedia contents.

These and other futures, aspects and advantages of the present invention will become understood with reference to the following description, appended claims, and accompanying figures.

DETAILED DESCRIPTION OF THE DRAWINGS

Referring now to the FIG. **1**, it demonstrates an example of a network infrastructure which includes a wireless device and a server. In the FIG. **1**, Net (**2**) represents a communication link, which may be combined with wireless and wired connection media and guarantee that the communication packets can be sent/received between the wireless device and the server. It is also assumed that the net (**2**) in the network infrastructure is built up in such way that a user from a web browser on a wireless device can access and browse any web-site on the Internet, and Intranet.

In the FIG. **1**, the console support software (**5**) of the server (**3**) can be configured to support web-based multitasking while a user on a wireless device (**1**) using a web browser (**8**). Further, the user of the wireless device is facilitated to perform creating structured layered file directories or folders, and perform data management operations, such as delete, move, copy, rename for data files or folders or directories etc. on an assigned storage volume controlled by the server (**3**).

In addition, the other software modules (**9**) of the wireless device (**1**) is also configured capable of to send data to or receive data from the other service modules (**7**) of the server (**3**) via communication link (**2**) through a suitable IP or non-IP based protocol. The data being sent or received cold be a digital photo picture, a message without limits.

Also, the console supporting software (**5**) of the server (**3**) and the other software modules (**9**) of the wireless device (**1**) can be implemented with any suitable languages such as C, C++, Java without limits.

Besides, the web-browser (**8**) of the wireless device (**1**) can be any suitable software, which. is capable to communication with web server software (**4**) on the server (**3**) or with any other web server through the HTTP protocol.

On the other hand, FIG. **2** has demonstrated the detailed storage system (**10**) of a server **3**, where its storage volumes can be allocated to multiple wireless devices as followings: First, the storage system (**10**) of the server (**3**) can be partitioned into multiple storage volumes (**11**), for example, by administration staff through a web-console (**13**) of a console host (**12**).

Second, the storage system (**10**) of the server (**3**) can be partitioned in such way that a user of each of the wireless devices can be assigned with a storage volume having a desired size, so that the server **3** can support maximum numbers of the wireless devices.

In addition, the storage connection media could be any kind, such as SCSI cable, IP cable, Fiber cable etc. or could be wireless communication media. The storage system itself could be various types.

Finally, the storage system (**10**) can be accessed by each of the wireless devices through IP or non-IP based network and protocols.

The FIG. **3** has demonstrated that a user from a web-browser (**8**) on a wireless device (**1**) can download data from a known web-site (**15**) to his/her assigned external storage (**10**) on the server (**3**). The dash-lined path (a) represents a communication link between the wireless device (**1**) and a remote web-site (**15**) that provides downloading contents. The dash-lined path (b) represents a communication link between the wireless devices (**1**) and the storage server (**3**). The dash-lined path (c) represents a communication link between the server **3** and the remote web-server (**15**).

THE DETAILED DESCRIPTION OF THE INVENTION

The Use of the External Storage by the Wireless Device:

The FIG. **2** shows a simplified diagram of the wireless devices (**1**) using the external storage system (**10**) of the server (**3**) for effectively resolving the storage limitation problem for the wireless devices (**1**).

Partition storage volumes (FIG. **2**):

With this invention, the entire storage (**10**) on the server (**3**) needs to be partitioned into suitable size of volumes (**11**) such as 4 GB for each volume. This will allow the server **3** to serve maximum number of the wireless devices (**1**). With the web console support software (**5**) of the server (**3**), tasks of partitioning the storage system (**10**) can be done through a web-console (**13**) on a console host (**12**) by an administrative staff.

In order to support storage partition, first the console support software (**5**) of the server (**3**) must send storage information of the server (**3**) to the web-console (**13**) of the console host (**12**). The storage information includes each storage device's name and total size etc. without limits. Second, based on the received storage information, the administration staff on the console host (**12**), for example, can use a web-console (**13**) to partition each storage device and send the storage partition information for the storage device to the console support software (**5**) of the server (**3**). The storage partition information includes the number of the partitions (volumes) and the size of each partition (volume). Third, upon receiving the storage partition information from the web-console (**13**) of the console host (**12**), the console support software (**5**) of the server (**3**) performs the actual storage partition by dividing the entire storage into multiple small volumes. Finally, for each small storage volume, a corresponding file system could be built up.

Assign storage volumes (FIG. **2**):

Each of the storage volumes (**11**) together with its corresponding file system on the storage system (**10**) of the server (**3**) needs to be exclusively assigned to a user of a specific wireless device (**1**) by the console support software (**5**) of the server (**3**).

Data and storage volume management (FIG. **2**):

With the support of the console support software modules (**5**) of the server (**3**) by following similar steps of said partitioning storage, the user of the wireless device (**1**) can utilize a web-browser **8** illustrated in FIG. **2** to setup folder or directory structure on the file system of his/her an assigned external storage volume (**11**). In addition, the user of the wireless device (**1**) can use the web-browser **8** performing all data management operations such as delete, copy, move, or rename for file or folder on the file system.

In order to support such data management over the external storage (**10**) assigned to the user of the wireless device (**1**) by using the web-browser **8**, first the console support software modules (**5**) of the server (**3**) must communicate with the web-browser (**8**) of the wireless device (**1**) to present the assigned storage to a user as described before. Therefore, the user from the web-browser (**8**) of the wireless device (**1**) can choose a desired data management operation and perform an operation causing the operation information to be sent to the console support software modules (**5**) of the server (**3**). The mentioned operation includes establishing folder or directory; copying, moving, or reaming data file etc. for the folder or directory. Second, upon receiving the data management operation, the console support software modules (**5**) of the server (**3**) actually performs these requested operations on the assigned file system of an assigned external storage volume (**11**) on the server (**3**).

Store data from wireless device into external storage (FIG. **2**):

To store the data such as digital photo pictures or messages into the file system on the assigned storage volume (**11**) in the server (**3**), the other software modules (**9**) of the wireless device (**1**) need to send these data to the other service modules (**7**) of the server (**3**) via communication link between them. Upon receiving the data, the other service modules (**7**) of the server (**3**) write these data to the file system of the assigned storage volume (**11**) on the server **3** for the wireless device **1**. The protocol used between these two communication entities could be either IP or non-IP based protocol.

Download data from a remote web server site into allocated storage volume:

Now, referring to the FIG. **3**, If a user of the wireless device (**1**) wants to download a data from a remote web server (**15**) into the file system on the assigned storage volume (**11**) in the external storage system (**10**) on the server (**3**), the following steps are required:

1) The user of the wireless device (**1**) via a web-browser (**8**) access to a remote web server site (**15**) to obtain information of the data for the downloading via the path (a) of the FIG. **3**. For example, the user access to a web-page which contains the data name for the downloading and also contains IP address of the remote web site.

2) The other software modules (**9**) of the wireless device (**1**) obtain the downloading information for the data, which becomes available in the cached web-pages on the wireless device (**1**) after the web-browser (**8**) access to the web site (**15**).

3) The other software modules (**9**) of the wireless device (**1**) send the obtained downloading information to other service modules (**7**) of the storage server (**3**) via the path (b) of the FIG. **3**.

4) Upon receiving the downloading information from the wireless device (**1**), the other service module (**7**) of the storage server (**3**) sends a web download request to the web-site (**15**) via the path (c) of the FIG. **3** based on download information obtained, and then receives the downloading data from the web server of the web-site (**15**).

5) Upon receiving downloading data streams, the other service modules (**7**) of the storage server (**3**) write the data into the file system on the assigned storage volume (**11**) in the server (**3**) for the wireless device (**1**).

Retrieve data from assigned storage volume for the user of the wireless device:

1) If a web-browser (**8**) on a wireless device **1** has embedded video or music functionality, a user of the wireless device (**1**) can use the browser to retrieve and play multimedia data file such as video or music stored in the assigned storage volume (**11**) located on the server (**3**).

2) In another embodiment, in respect to the user's needs, the other software module (**9**) of the wireless device (**1**) also can retrieve data file from the file system of the assigned storage volume (**11**) on the server (**3**).

Support external storage for a large number of the wireless devices:

If there is a need to provide each user a 2 GB of storage space, then a 160 GB disk drive can support 80 users. A 4096 GB (4 Tera Bytes) storage system on the server unit can support 2024 user. Each of the server units only can efficiently support a limited size of the storage system. In order to support a large number of the wireless devices, such as for 500,000 wireless devices, a larger number of the servers is required, in this case 250 servers is required. In order to let a larger number of the servers to effectively support the larger number of the wireless devices, an infrastructure like the CCDSVM is desirable, which has been described in prior patent applications. With the CCDSVM the control management system can control larger number of storage servers to provide external storage for a huge number of the wireless devices.

The invention claimed is:

1. A server for delivering storage service, comprising:

a plurality of storage spaces; and

a non-transitory computer-readable medium comprising program instructions that, being executed by the server, causes the server delivering the storage service; wherein the program instructions comprise:

program instructions for allocating exclusively a first one of the storage spaces to a user of a first wireless device;

program instructions for establishing a communication link for the first wireless device remotely access to the first one of the storage spaces;

program instructions for presenting the first one of the storage spaces to the user on the first wireless device through communication with the first wireless device; and

program instructions for updating the first one of the storage spaces according to a requested operation for remotely access to the first one of the storage spaces in response to the user from the first wireless device performing the operation,

wherein said operation comprises storing data into the first one of the storage spaces or retrieving data therefrom,

the storing of said data including to download a file from a remote server across a network into the first one of the storage spaces through utilizing download information for the file cached in the first wireless device in response to the user from the first wireless device performing the operation for downloading the file.

2. The server as recited in claim **1**, wherein said downloading a file from a remote server further comprises:

the first wireless device obtaining downloading information for the file from the remote server, transmitting the cached downloading information to the server, and causing the server in accordance with the downloading information to download the file into the first one of the storage spaces.

3. The server as recited in claim **1**, wherein said storage spaces further are configured among a plurality of storage devices.

4. The server as recited in claim **1**, wherein the data being stored into or retrieved from the first one of the storage spaces, further is a message or multimedia data of video, digital music, or digital picture.

5. The server as recited in claim **1**, wherein said operation further comprises:

from the first wireless device remotely deleting, moving, copying or renaming a folder in the first one of the storage spaces.

6. The server as recited in claim **5**, wherein said operation further comprises:

from the first wireless device remotely deleting, moving, copying or renaming a file in the first one of the storage spaces.

7. The server as recited in claim **1**, wherein said operation further comprises:

from the first wireless device remotely creating a folder or a folder structure in the first one of the storage spaces.

8. The server as recited in claim **1**, wherein said program instructions further configures the server to cause display of the first one of the storage spaces in a web browser executed on the first wireless device for the user remotely access to the first one of the storage spaces.

9. The server as recited in claim **1**, wherein the program instructions further causes the server to allocate exclusively a second one of the storage spaces of a predefined capacity to a user of a second wireless device for facilitating the user remotely access to the second one of the storage spaces.

10. The server as recited in claim **1**, wherein the first wireless device further is one of a cell phone or a personal data assistant and management device ("PDA").

11. The server as recited in claim **1**, wherein the download information for the file further includes at least the name of the file and the internet protocol ("IP") address of the remote server.

12. A server for delivering storage service, comprising:

a plurality of storage spaces, and a non-transitory computer-readable medium comprising program instructions that, executed by the server, causes the server to deliver the storage service; wherein the program instructions comprises:

program instructions for the server allocating exclusively a first one of the storage spaces of a predefined capacity to a user of a first wireless device;

program instructions for establishing a communication link for the first wireless device remotely access to the first one of the storage spaces;

program instructions for sending information of the first one of the storage spaces to the first wireless device for presenting the first one of the storage spaces to the user; and

program instructions for updating the first one of the storage spaces according to a requested operation for remotely access to the first one of the storage spaces in response to the user from the first wireless device performing the operation,

wherein said operation comprises creating from the first wireless device a folder structure of a plurality of folders in the first one of the storage spaces, and comprises to delete or move or copy or rename a first one of the folders in the folder structure, wherein each of the folders being used by the first wireless device for storing data therein or retrieving data therefrom.

13. The server as recited in claim **12**, wherein said storing data includes to download a file from a remote server across a network into the first one of the storage spaces through utilizing download information for the file cached in the first wireless device in response to the user from the first wireless device performing the operation for downloading the file.

14. The server as recited in claim **13**, wherein said downloading a file comprises: the first wireless device obtaining download information for the file from the remote server, transmitting the cached download information to the server; and causing the server in accordance with the downloading information to store the file from the remote server into the first one of the storage spaces.

15. The server as recited in claim **12**, wherein said operation further comprises: from the first wireless device to create a folder in the folder structure.

16. The server as recited in claim **12**, wherein said data being stored into or retrieved from the first one of the storage spaces, further comprises a message, or a multimedia data of video, digital music or photo picture.

17. The server as recited in claim **12**, wherein said operation further comprises: from the first wireless device to delete, move, copy or rename a file in the folder structure.

18. The server as recited in claim **12** wherein said storage spaces further are configured among a plurality of storage devices.

19. The server as recited in claim **13**, wherein said download information for the file further includes at least the name of the file and the internet protocol ("IP") address of the remote server.

20. The server as recited in claim **12**, wherein the program instructions further causes the server to: allocate exclusively a second one of the storage spaces of a predefined capacity to a user of a second wireless device for facilitating the user remotely access to the second one of the storage spaces.

* * * * *

EXHIBIT H

US007870225B2

(12) **United States Patent**
Kim

(10) **Patent No.: US 7,870,225 B2**
(45) **Date of Patent: Jan. 11, 2011**

(54) **DISK SYSTEM ADAPTED TO BE DIRECTLY ATTACHED TO NETWORK**

(75) Inventor: **Han-gyoo Kim**, Seoul (KR)

(73) Assignee: **Zhe Khi Pak**, Moscow (RU)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **12/701,335**

(22) Filed: **Feb. 5, 2010**

(65) **Prior Publication Data**

US 2010/0138602 A1 Jun. 3, 2010

Related U.S. Application Data

(62) Division of application No. 09/974,082, filed on Oct. 9, 2001, now Pat. No. 7,792,923.

(60) Provisional application No. 60/240,344, filed on Oct. 13, 2000.

(51) **Int. Cl.**
G06F 15/16 (2006.01)
G06F 3/00 (2006.01)
G06F 12/00 (2006.01)

(52) **U.S. Cl.** **709/217**; 709/218; 709/236; 709/246; 709/250; 710/5; 710/36; 711/110

(58) **Field of Classification Search** 709/217, 709/218, 236, 246, 250; 711/110; 710/5
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,329,619 A 7/1994 Page et al.
5,426,427 A 6/1995 Chinnock et al.
5,455,926 A 10/1995 Keele et al. 711/4
5,459,857 A * 10/1995 Ludlam et al. 714/6
5,463,772 A 10/1995 Thompson et al.
5,513,314 A 4/1996 Kandasamy et al.

(Continued)

FOREIGN PATENT DOCUMENTS

DE 19610840 9/1997

(Continued)

OTHER PUBLICATIONS

EP Communication 94(3) issued in co-pending European Application No. 01 272 932.3-2413 (Issued Aug. 3, 2009) (9 pages).

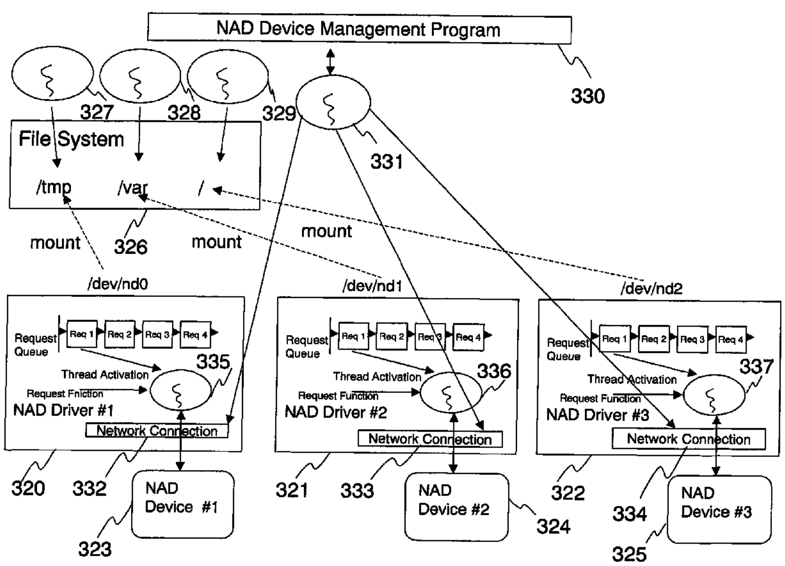
(Continued)

Primary Examiner—Saleh Najjar
Assistant Examiner—Vitali Korobov
(74) *Attorney, Agent, or Firm*—Rothwell, Figg, Ernst & Manbeck, P.C.

(57) **ABSTRACT**

A network-attached disk (NAD) system is disclosed that includes an NAD device for receiving a disk access command from a host through a network, and a device driver at the host for controlling the NAD device through the network, where the device driver creates a virtual host bus adapter so that the host recognizes the NAD device as if it is a local device to the host. The host may run the UNIX or Windows family of operating systems. The NAD device includes a disk for storing data, a disk controller for controlling the disk, and a network adapter for receiving a disk access command from the host through a network port.

22 Claims, 24 Drawing Sheets



U.S. PATENT DOCUMENTS

5,524,247	A	6/1996	Mizuno	
5,566,331	A *	10/1996	Irwin et al.	1/1
5,642,337	A	6/1997	Oskay et al.	709/230
5,721,818	A	2/1998	Hanif et al.	
5,774,660	A	6/1998	Brendel et al.	
5,781,550	A	7/1998	Templin et al.	
5,812,930	A	9/1998	Zavrel	
5,838,916	A	11/1998	Domenikos et al.	
5,845,104	A	12/1998	Rao	711/113
5,889,942	A	3/1999	Orenshteyn	
5,941,972	A *	8/1999	Hoese et al.	710/315
5,987,523	A	11/1999	Hind et al.	
5,987,627	A	11/1999	Raqlings, III	
5,999,808	A	12/1999	Ladue	
6,047,307	A	4/2000	Radko	
6,085,234	A	7/2000	Pitts et al.	
6,128,644	A	10/2000	Nozaki	
6,128,690	A	10/2000	Purcell et al.	
6,167,490	A	12/2000	Levy et al.	
6,175,869	B1	1/2001	Ahuja et al.	
6,216,202	B1 *	4/2001	D'Errico	711/112
6,314,465	B1	11/2001	Paul et al.	
6,317,775	B1	11/2001	Colie et al.	
6,327,594	B1	12/2001	Van Huben et al.	
6,345,300	B1	2/2002	Bakashi et al.	
6,347,095	B1	2/2002	Tang et al.	
6,351,776	B1 *	2/2002	O'Brien et al.	709/245
6,356,915	B1 *	3/2002	Chtchetkine et al.	707/823
6,360,265	B1	3/2002	Falck et al.	
6,366,988	B1 *	4/2002	Skiba et al.	711/165
6,389,432	B1 *	5/2002	Pothapragada et al.	1/1
6,393,569	B1	5/2002	Orenshteyn	
6,404,766	B1	6/2002	Kitai et al.	
6,421,753	B1 *	7/2002	Hoese et al.	710/315
6,449,647	B1	9/2002	Colby et al.	
6,470,389	B1	10/2002	Chung et al.	
6,510,164	B1	1/2003	Ramaswamy et al.	
6,518,965	B2	2/2003	Dye et al.	
6,523,066	B1	2/2003	Montroy et al.	
6,529,996	B1 *	3/2003	Nguyen et al.	711/114
6,539,446	B1	3/2003	Chan	
6,578,111	B1	6/2003	Damron et al.	
6,594,677	B2	7/2003	Davis et al.	
6,598,068	B1	7/2003	Clark	
6,609,167	B1	8/2003	Bastiani et al.	
6,647,016	B1	11/2003	Isoda et al.	
6,732,104	B1	5/2004	Weber	
6,760,783	B1	7/2004	Berry	
6,807,581	B1 *	10/2004	Starr et al.	709/250
6,823,458	B1	11/2004	Lee et al.	
6,834,326	B1 *	12/2004	Wang et al.	711/114
6,894,981	B1	5/2005	Coile et al.	
6,941,576	B2	9/2005	Amit	
6,985,927	B2 *	1/2006	O'Brien et al.	709/213
7,010,303	B2	3/2006	Lewis et al.	
7,069,312	B2	6/2006	Kostic et al.	
7,069,350	B2	6/2006	Fujita et al.	
7,076,690	B1	7/2006	Todd et al.	
7,124,128	B2	10/2006	Springer et al.	
7,251,704	B2	7/2007	Solomon et al.	
7,254,578	B2	8/2007	Devarakonda et al.	
7,277,955	B2	10/2007	Elliott	
7,376,133	B2	5/2008	Gettala et al.	
7,383,229	B2	6/2008	Jacoby	
2001/0044879	A1 *	11/2001	Moulton et al.	711/114
2002/0103889	A1 *	8/2002	Markson et al.	709/223
2003/0014569	A1	1/2003	Kim	
2003/0018403	A1	1/2003	Braun et al.	
2003/0028614	A1	2/2003	Jeon	
2003/0172149	A1	9/2003	Edsall et al.	
2003/0225834	A1	12/2003	Lee et al.	
2004/0068563	A1	4/2004	Ahuja et al.	
2004/0117813	A1	6/2004	Karaoguz et al.	
2004/0220933	A1	11/2004	Walker	
2005/0042591	A1	2/2005	Bloom et al.	
2005/0110768	A1	5/2005	Marriott et al.	
2005/0149682	A1	7/2005	Kim	
2005/0193017	A1	9/2005	Kim	
2005/0193189	A1	9/2005	Kim	
2006/0004935	A1	1/2006	Seto et al.	
2006/0010287	A1	1/2006	Kim	
2006/0045130	A1	3/2006	Kim	
2006/0067356	A1	3/2006	Kim	
2006/0069884	A1	3/2006	Kim	
2006/0155805	A1	7/2006	Kim	
2007/0008988	A1	1/2007	Kim	

FOREIGN PATENT DOCUMENTS

DE	10038142	2/2001
JP	10-113469	5/1998
JP	10-271562	10/1998
JP	11007404	1/1999
JP	11-114224	4/1999
KR	10-2000-72493	12/2000
KR	10-2001-0088528	9/2001
WO	WO99/03297	7/1999
WO	WO00/29529	5/2000
WO	WO 00/29529 A2	5/2000

OTHER PUBLICATIONS

Yaron Klein, Sanrad, "Storage Virtualization with iSCSI Protocol; draft-klein-ips-virt-00.txt" (Nov. 2, 2000) (15 pages).

Blunden, Mark, et al, "Storage Networking Virtualization What's it all about?" IBM Redbooks (Dec. 2000) (124 pages).

Schulz,Greg, "SAN and NAS; Complementary Technologies—SAN and NAS provide Storage and Data Sharing" Internet Citation (May 1, 2000) (11 pages).

Blunden et al., "Storage Network Virtualization: What's it all about?", ibm.com/redbooks, XP-002286341, pp. 1-110, Dec. 2000.

Klein, Yaron, "Storage Virtualization with iSCSI Protocol", Internet Draft, XP-015030964, pp. 1-15, Nov. 2, 2000.

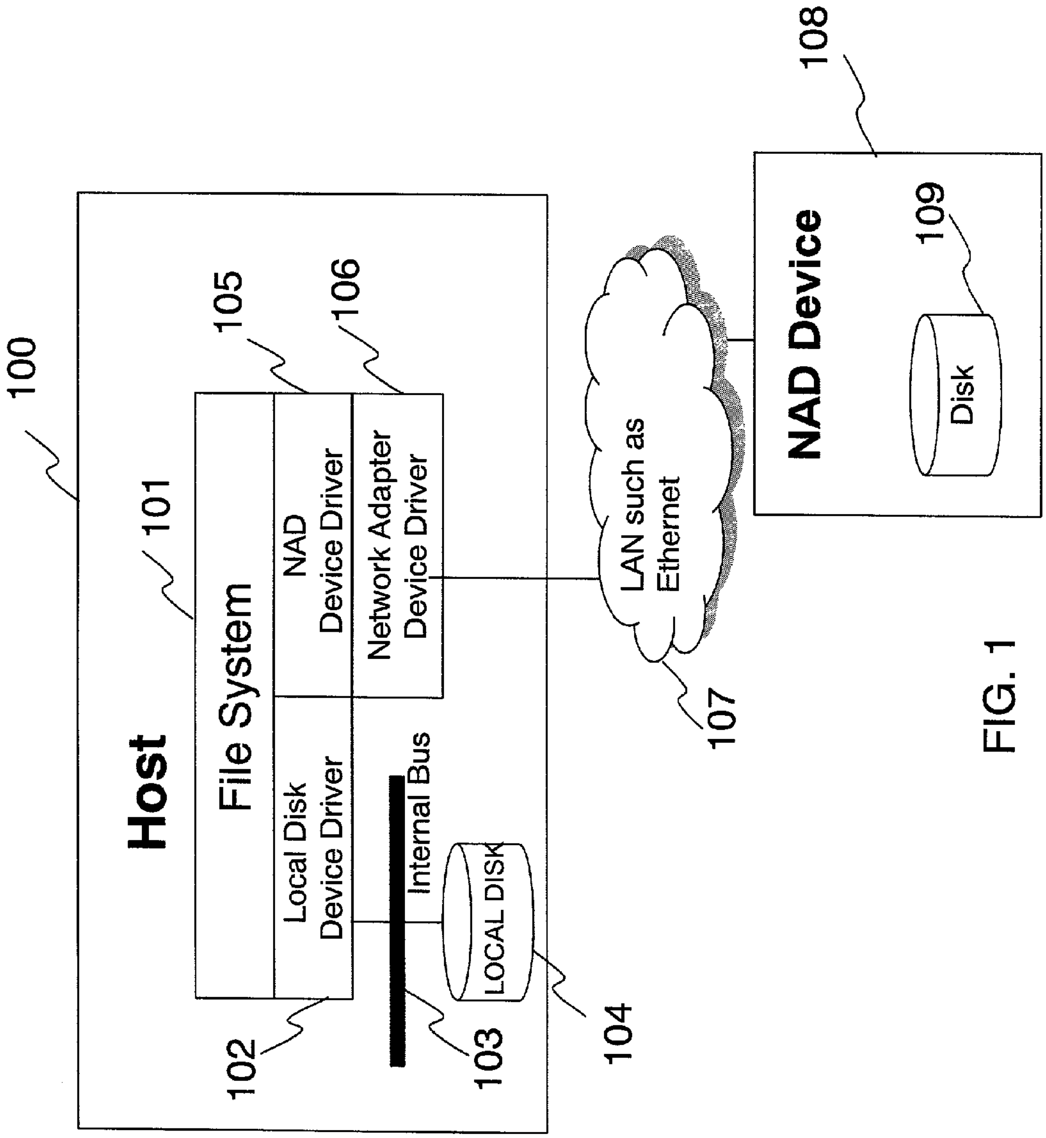
Schulz, Greg, "SAN and NAS; Complementary Technologies", http://www.mti.com/white_papers/WP20002.pdf, XP-002201566, pp. 1-7, May 1, 2000.

Supplementary European Search Report, Application No. 01272932. 3, 4 pages, Nov. 5, 2007.

Japanese Office Action, App. No. 555298/2002, Jan. 9, 2007.

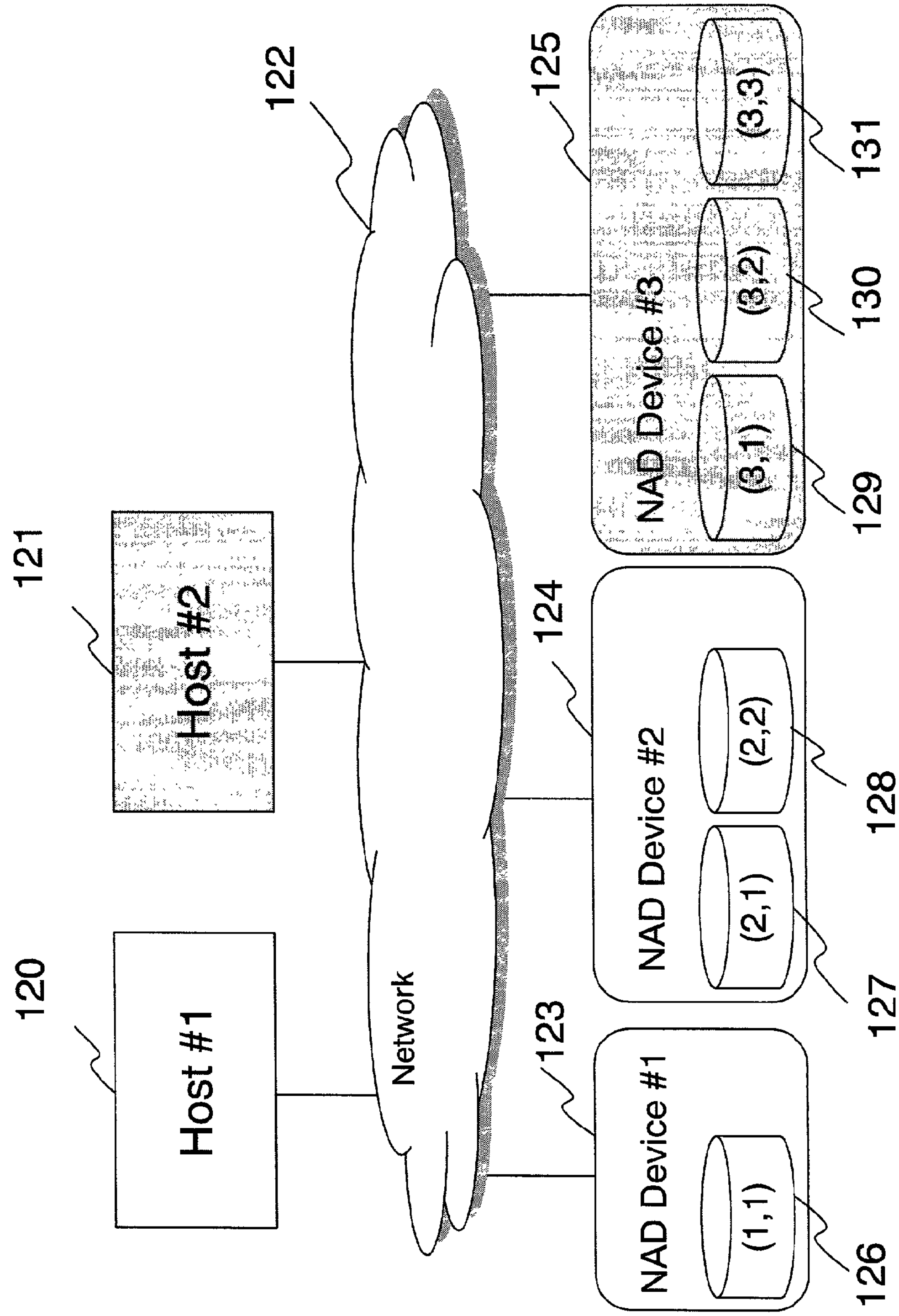
Japanese Office Action, App. No. 513300/2003, Feb. 6, 2007.

* cited by examiner



100
Host
101
File System
105
NAD
Device Driver
106
Network Adapter
Device Driver
Local Disk
Device Driver
102
Internal Bus
103
LOCAL DISK
104
LAN such as
Ethernet
107
108
NAD Device
109
Disk

FIG. 1



120
Host #1
121
Host #2
122
Network
123
NAD Device #1
(1,1)
126
124
NAD Device #2
(2,1)
(2,2)
127
128
125
NAD Device #3
(3,1)
(3,2)
(3,3)
129
130
131

FIG. 2

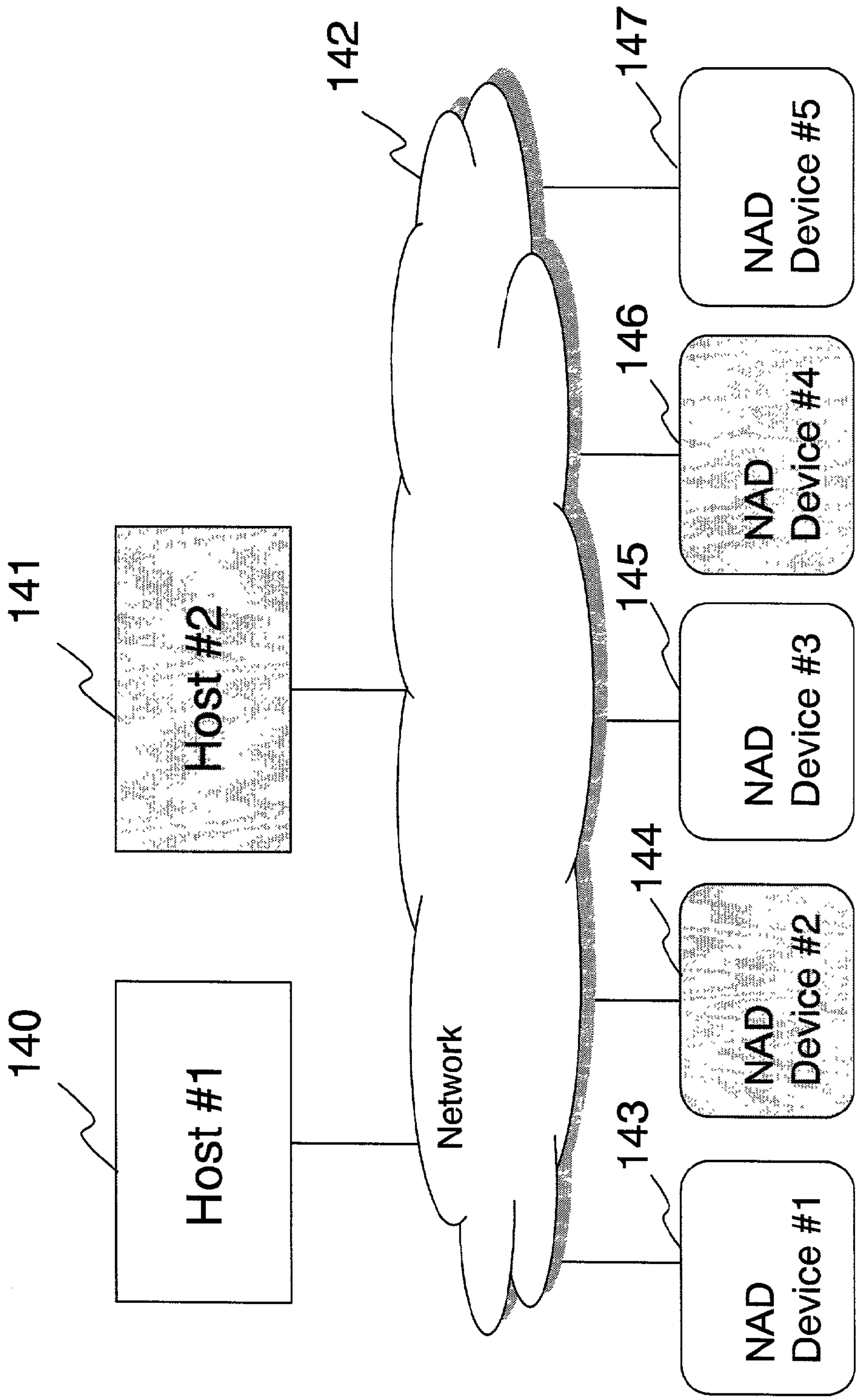

140
Host #1
141
Host #2
142
Network
143
NAD Device #1
144
NAD Device #2
145
NAD Device #3
146
NAD Device #4
147
NAD Device #5

FIG. 3

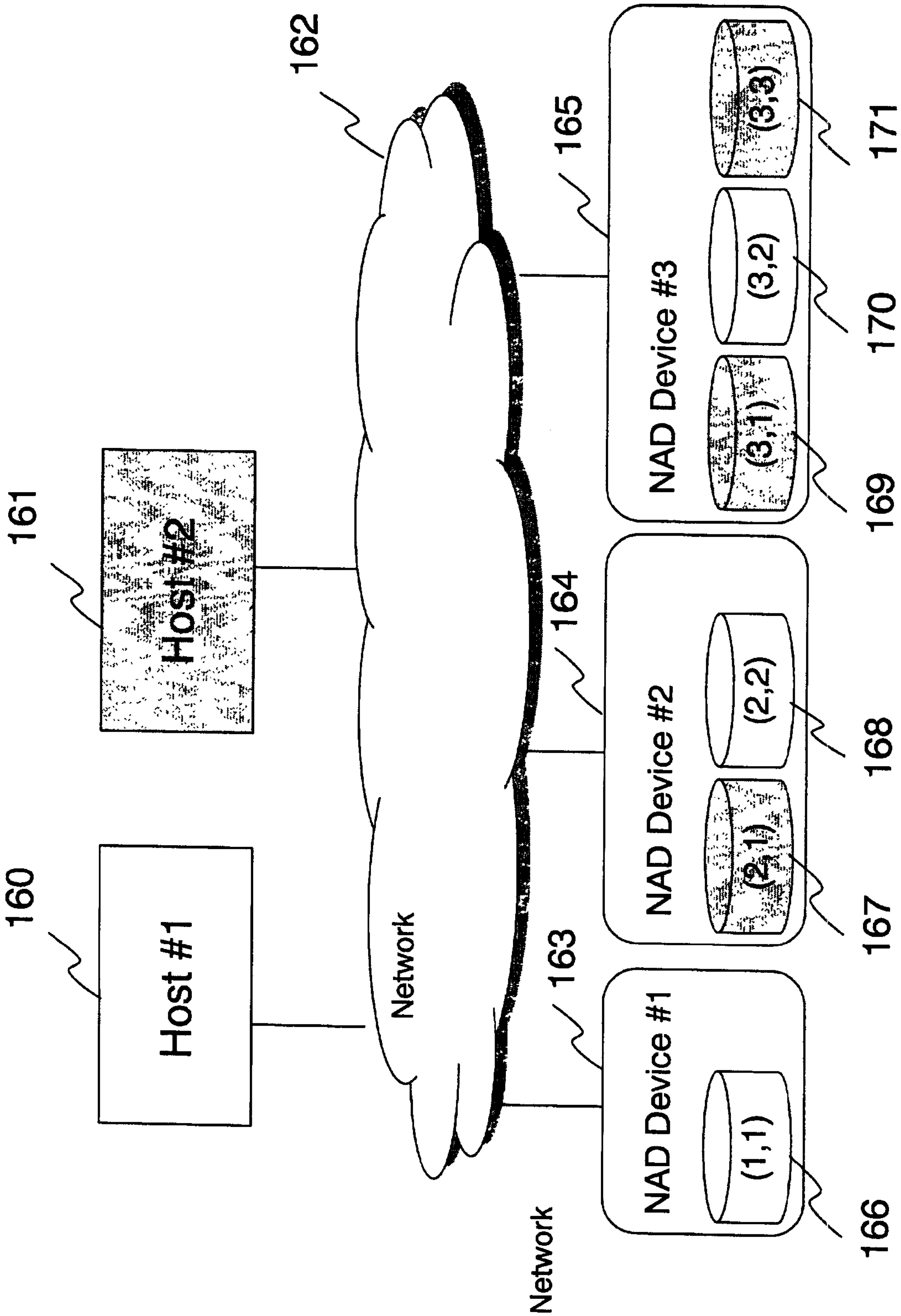
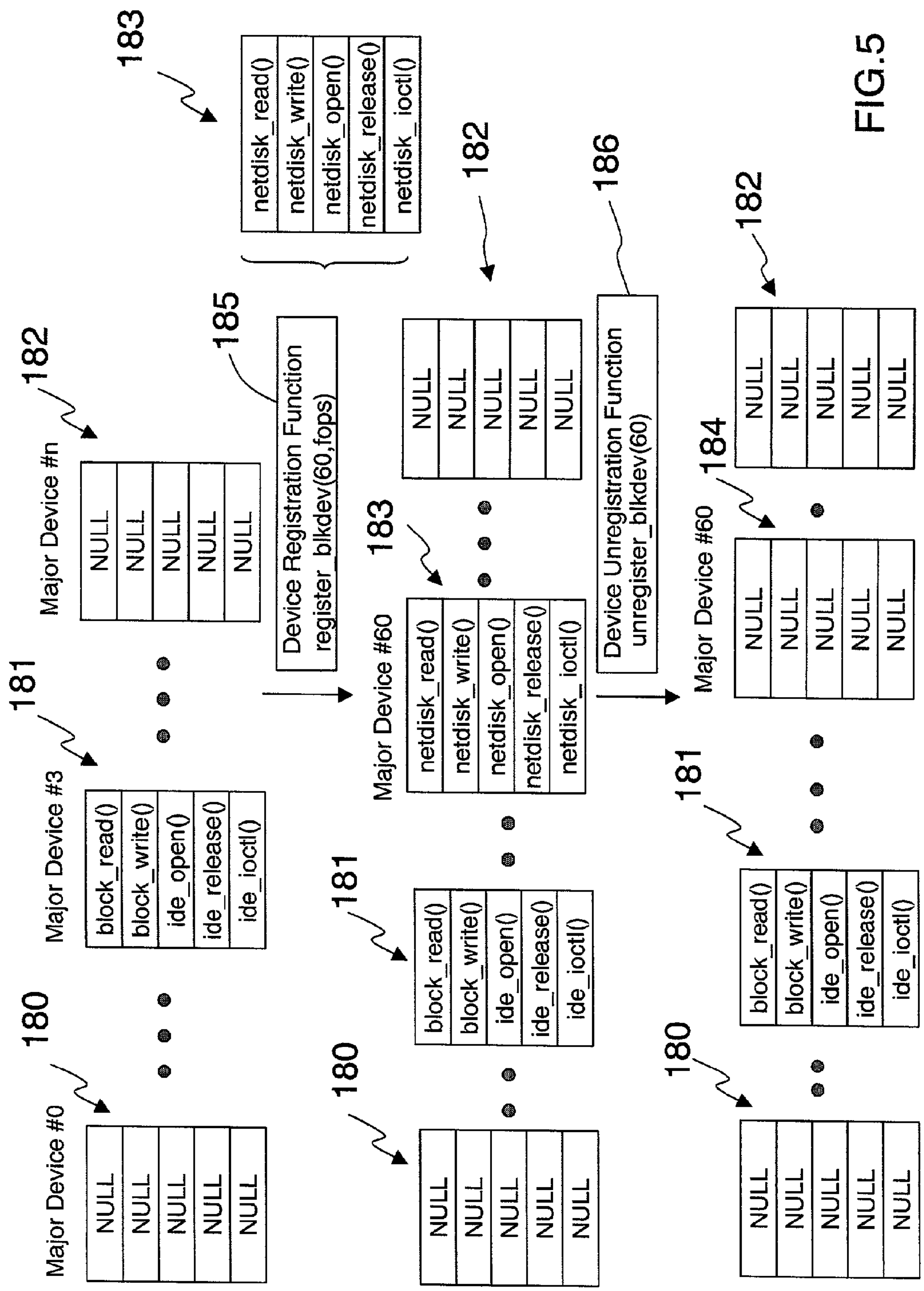

Host #1
Host #2
Network
Network
NAD Device #1
NAD Device #2
NAD Device #3
(1,1)
(2,1)
(2,2)
(3,1)
(3,2)
(3,3)
160
161
162
163
164
165
166
167
168
169
170
171

FIG. 4



Major Device #0
NULL
NULL
NULL
NULL
NULL
180
Major Device #3
block_read()
block_write()
ide_open()
ide_release()
ide_ioctl()
181
Major Device #n
NULL
NULL
NULL
NULL
NULL
182
183
netdisk_read()
netdisk_write()
netdisk_open()
netdisk_release()
netdisk_ioctl()
185
Device Registration Function
register_blkdev(60,fops)
180
181
Major Device #60
netdisk_read()
netdisk_write()
netdisk_open()
netdisk_release()
netdisk_ioctl()
183
182
186
Device Unregistration Function
unregister_blkdev(60)
180
181
Major Device #60
184
182

FIG.5

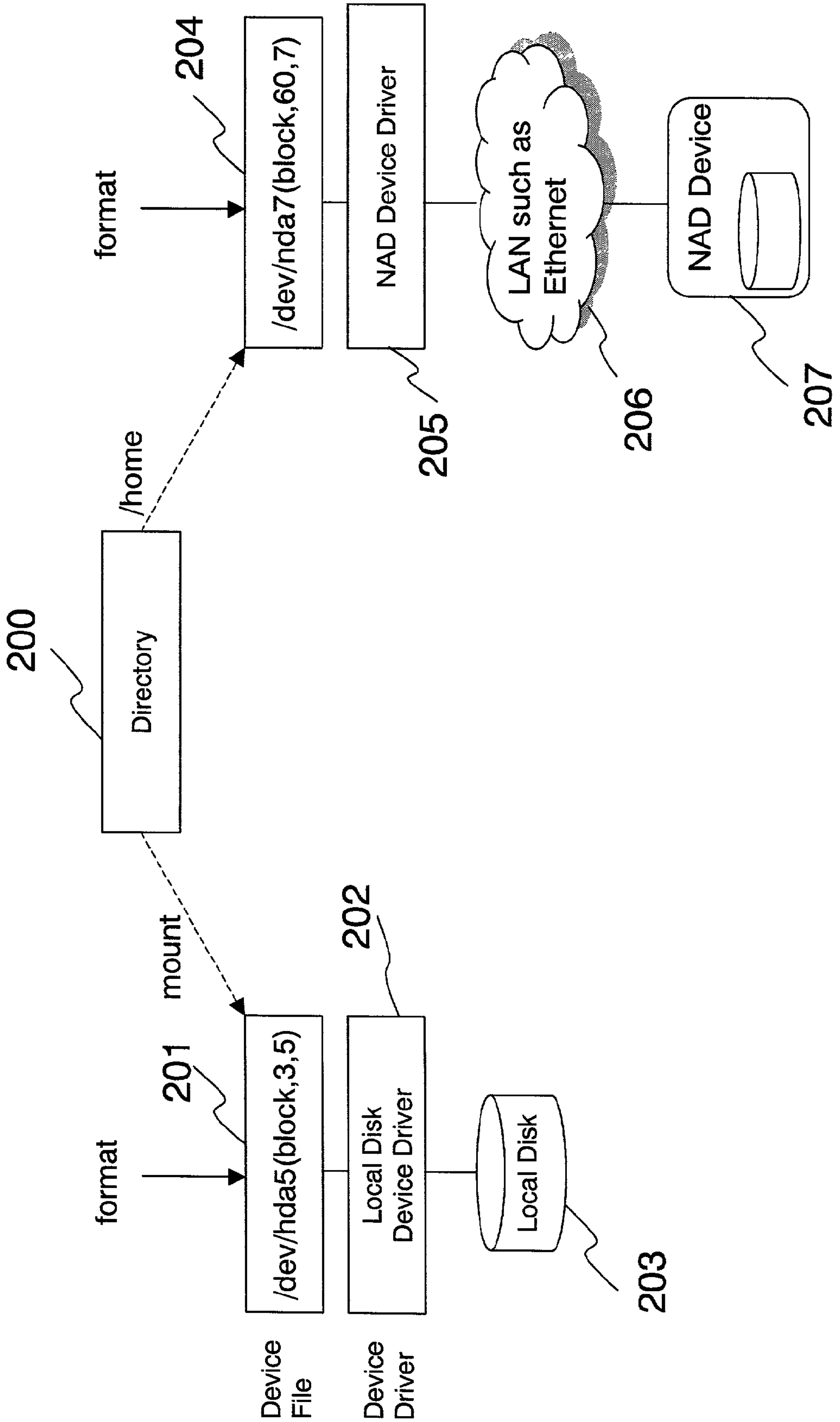

Device File
Device Driver
format
/dev/hda5(block,3,5)
201
Local Disk Device Driver
202
Local Disk
203
mount
Directory
200
/home
format
/dev/nda7(block,60,7)
204
NAD Device Driver
205
LAN such as Ethernet
206
NAD Device
207

FIG. 6

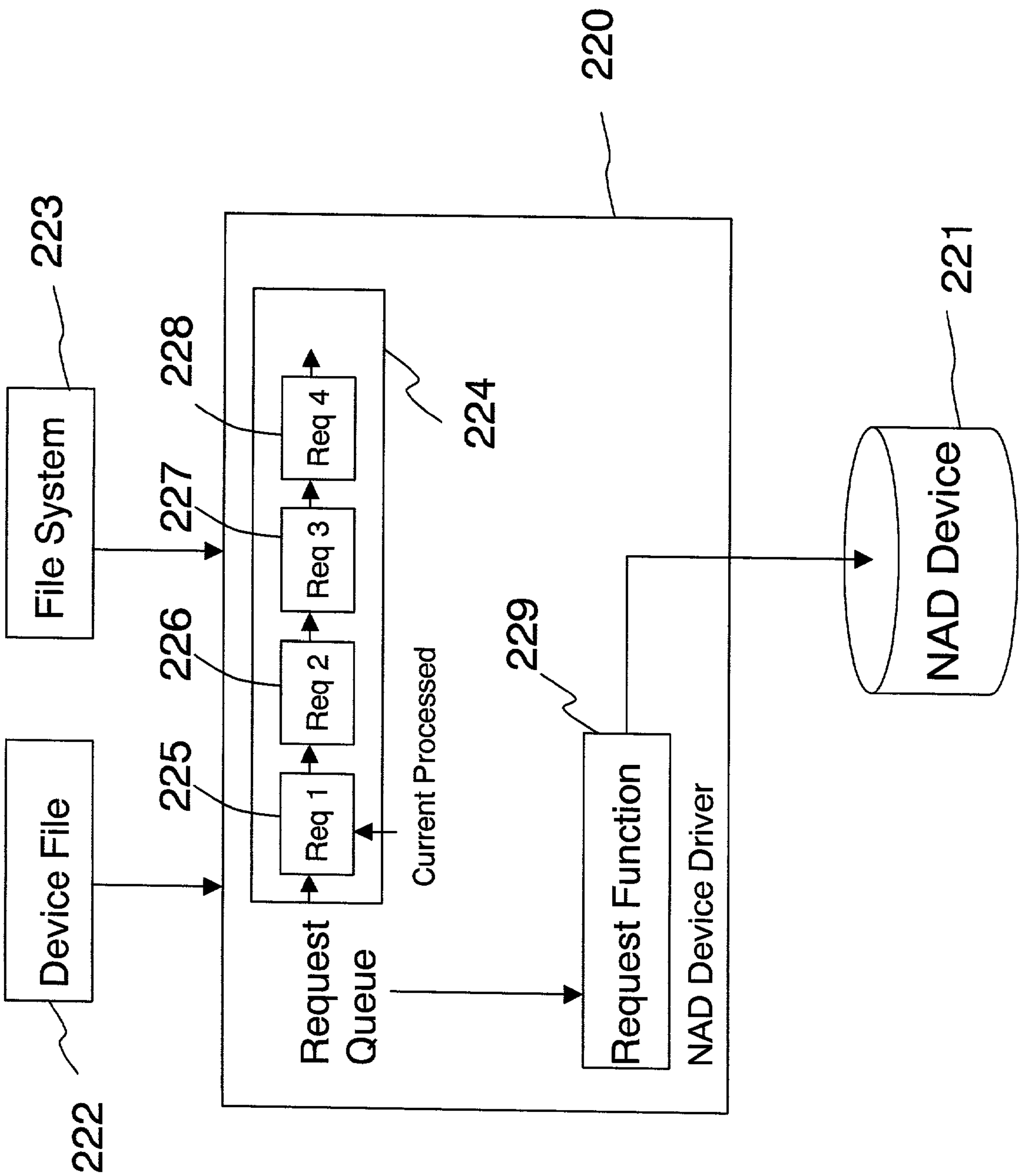

220
223
File System
Device File
222
228
227
226
225
Req 4
Req 3
Req 2
Req 1
224
Current Processed
Request
Queue
229
Request Function
NAD Device Driver
221
NAD Device

FIG. 7

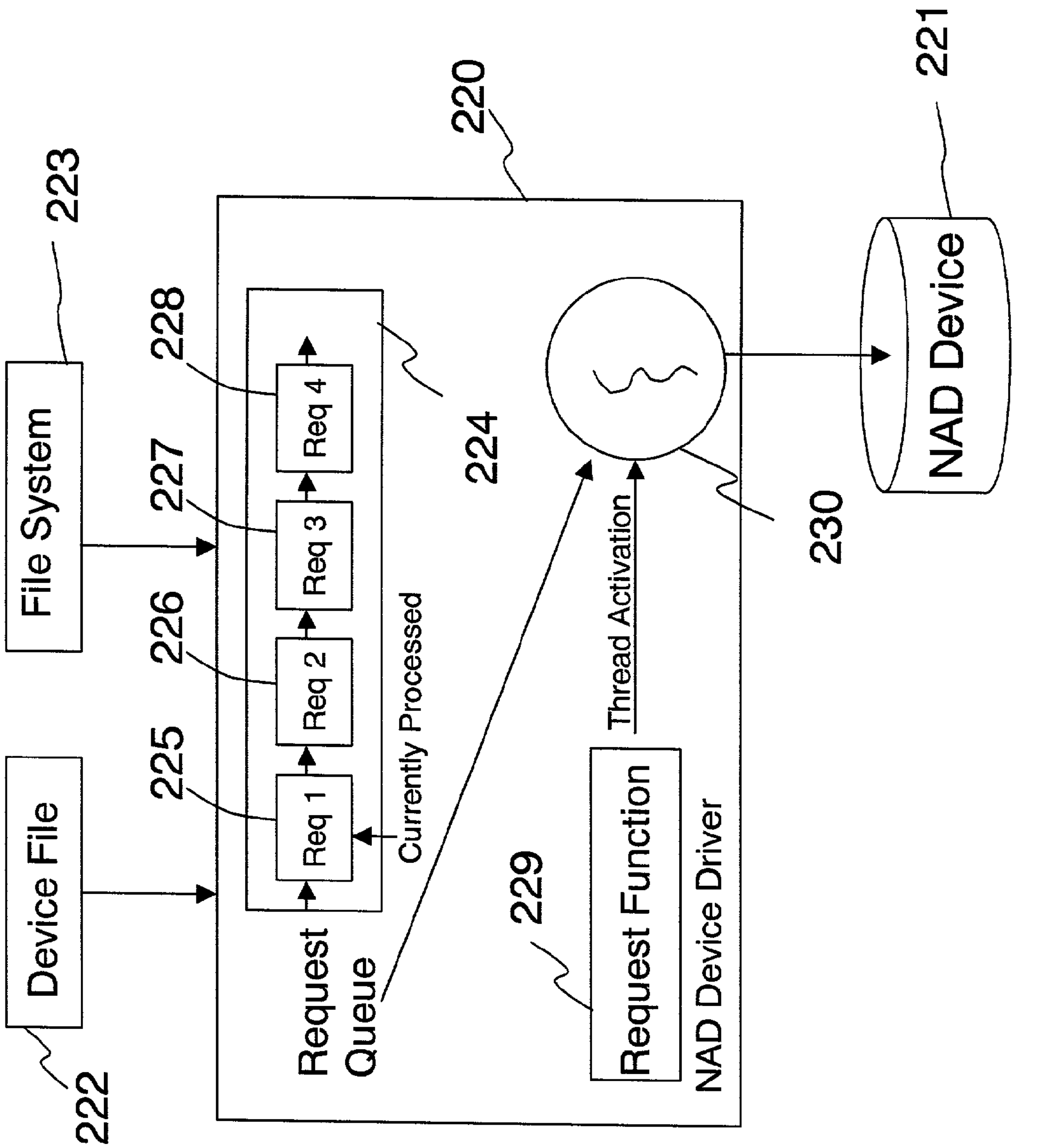

File System
Device File
Request Queue
Req 1
Req 2
Req 3
Req 4
Currently Processed
Thread Activation
Request Function
NAD Device Driver
NAD Device
220
221
222
223
224
225
226
227
228
229
230

FIG. 8

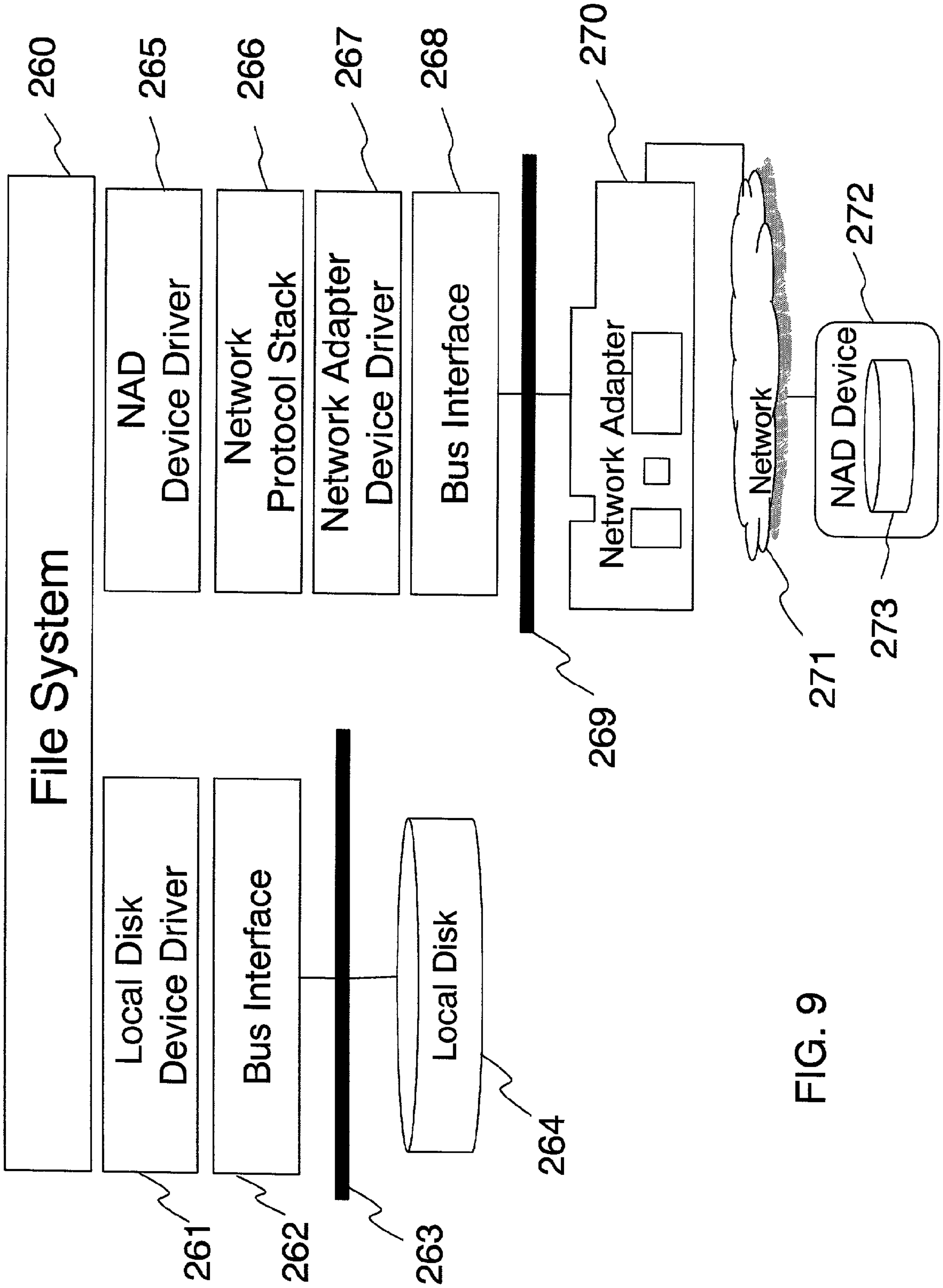

File System
260
NAD Device Driver
265
Network Protocol Stack
266
Network Adapter Device Driver
267
Bus Interface
268
270
Network Adapter
269
Network
271
NAD Device
272
273
Local Disk Device Driver
261
Bus Interface
262
263
Local Disk
264

FIG. 9

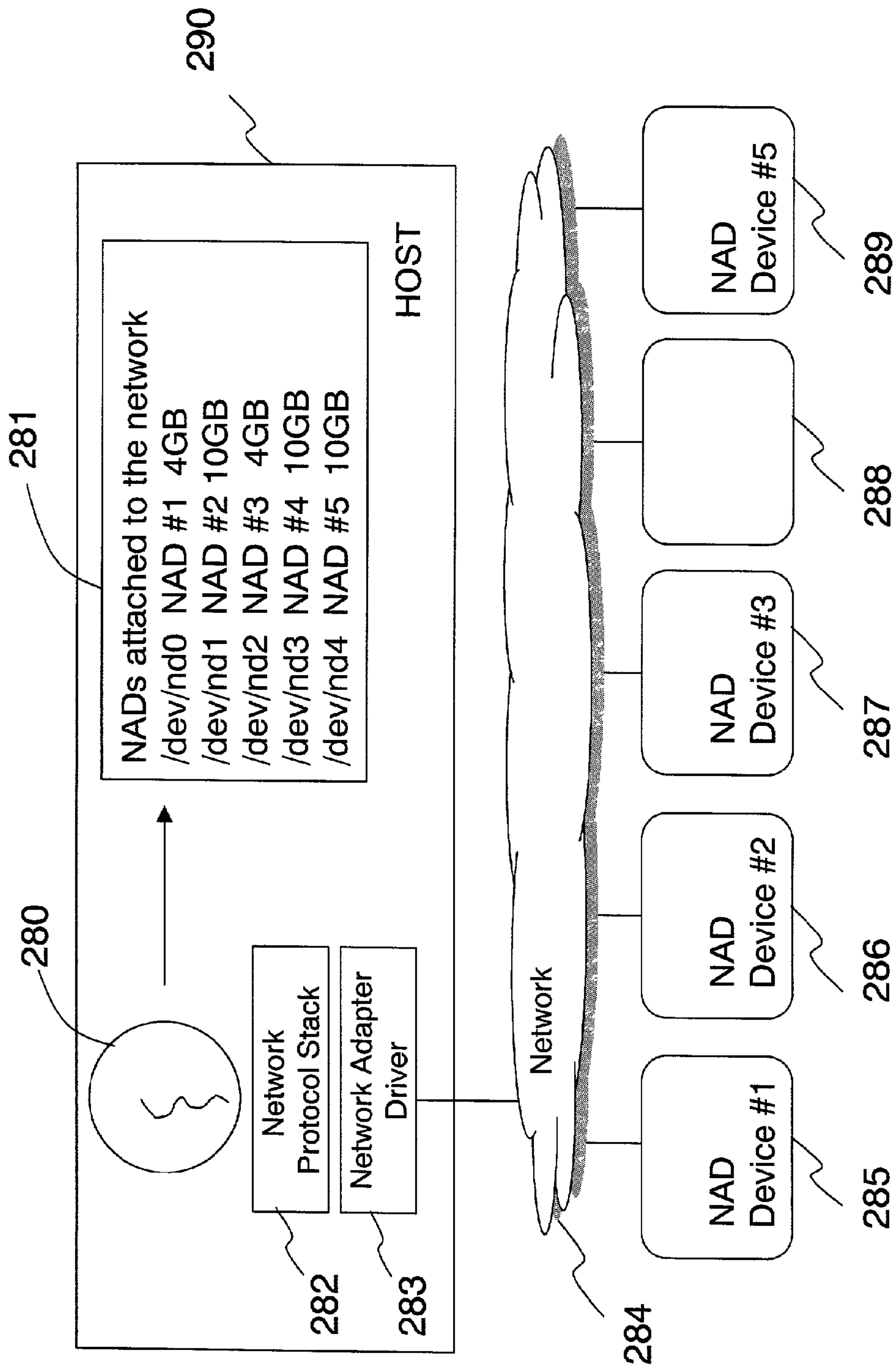

290
281
280
NADs attached to the network
/dev/nd0 NAD #1 4GB
/dev/nd1 NAD #2 10GB
/dev/nd2 NAD #3 4GB
/dev/nd3 NAD #4 10GB
/dev/nd4 NAD #5 10GB
HOST
Network Protocol Stack
Network Adapter Driver
282
283
Network
284
NAD Device #1
NAD Device #2
NAD Device #3
NAD Device #5
285
286
287
288
289

FIG. 10

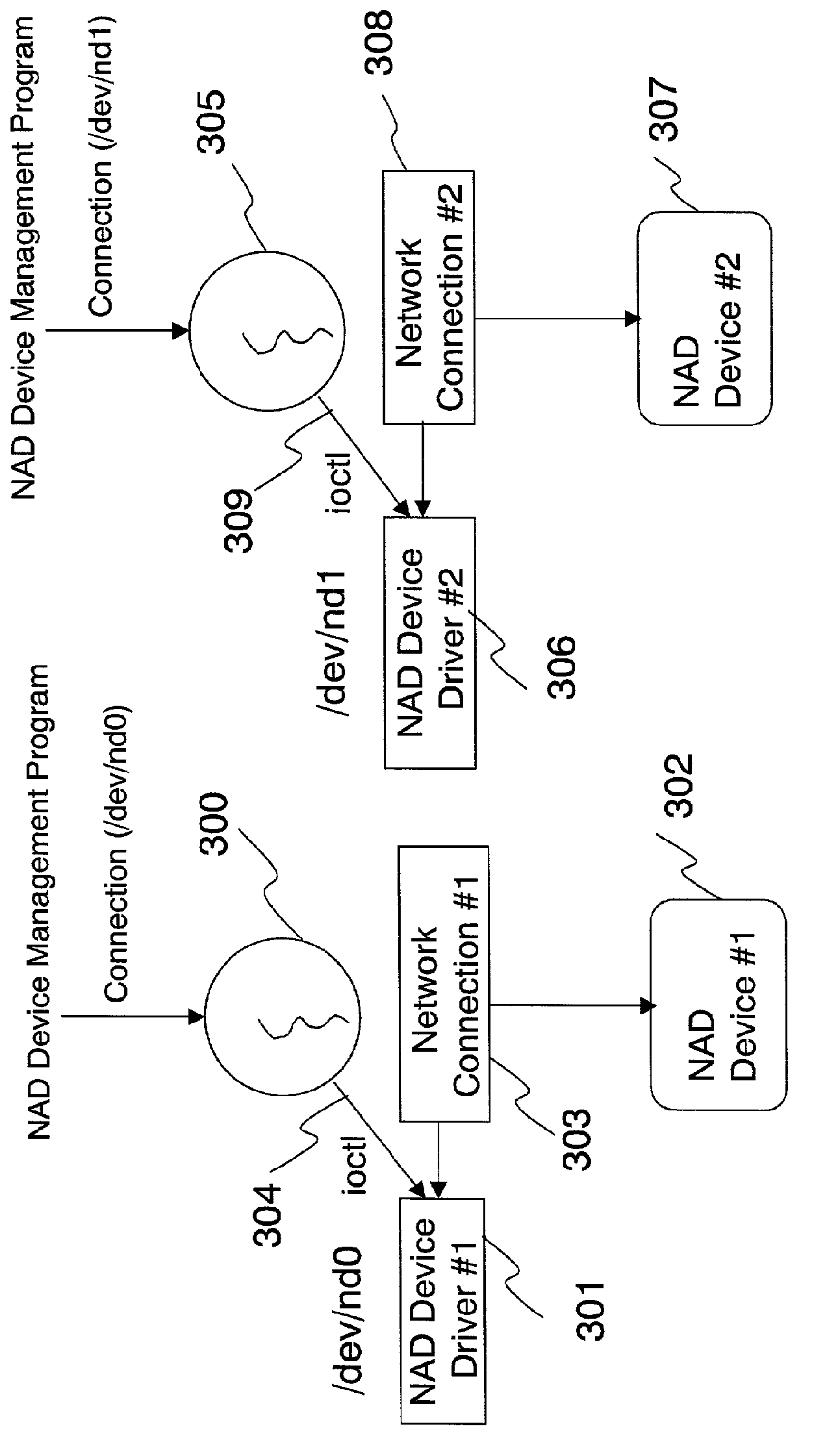


FIG. 11B

FIG. 11A

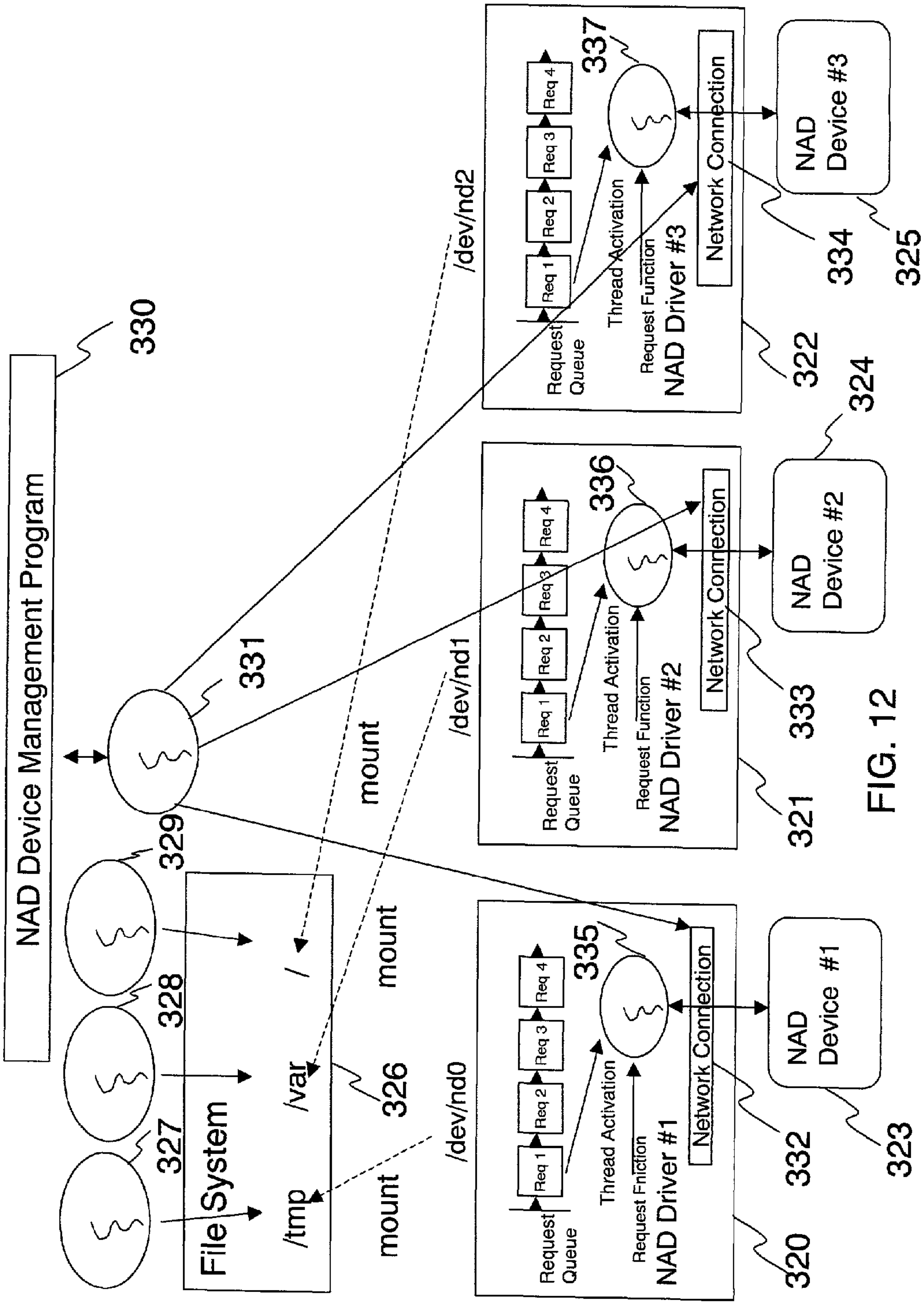

NAD Device Management Program
330
331
327
328
329
File System
/tmp
/var
/
326
mount
mount
mount
/dev/nd0
/dev/nd1
/dev/nd2
Request Queue
Req 1
Req 2
Req 3
Req 4
Thread Activation
Request Function
NAD Driver #1
NAD Driver #2
NAD Driver #3
Network Connection
335
336
337
NAD Device #1
NAD Device #2
NAD Device #3
320
321
322
323
324
325
332
333
334

FIG. 12

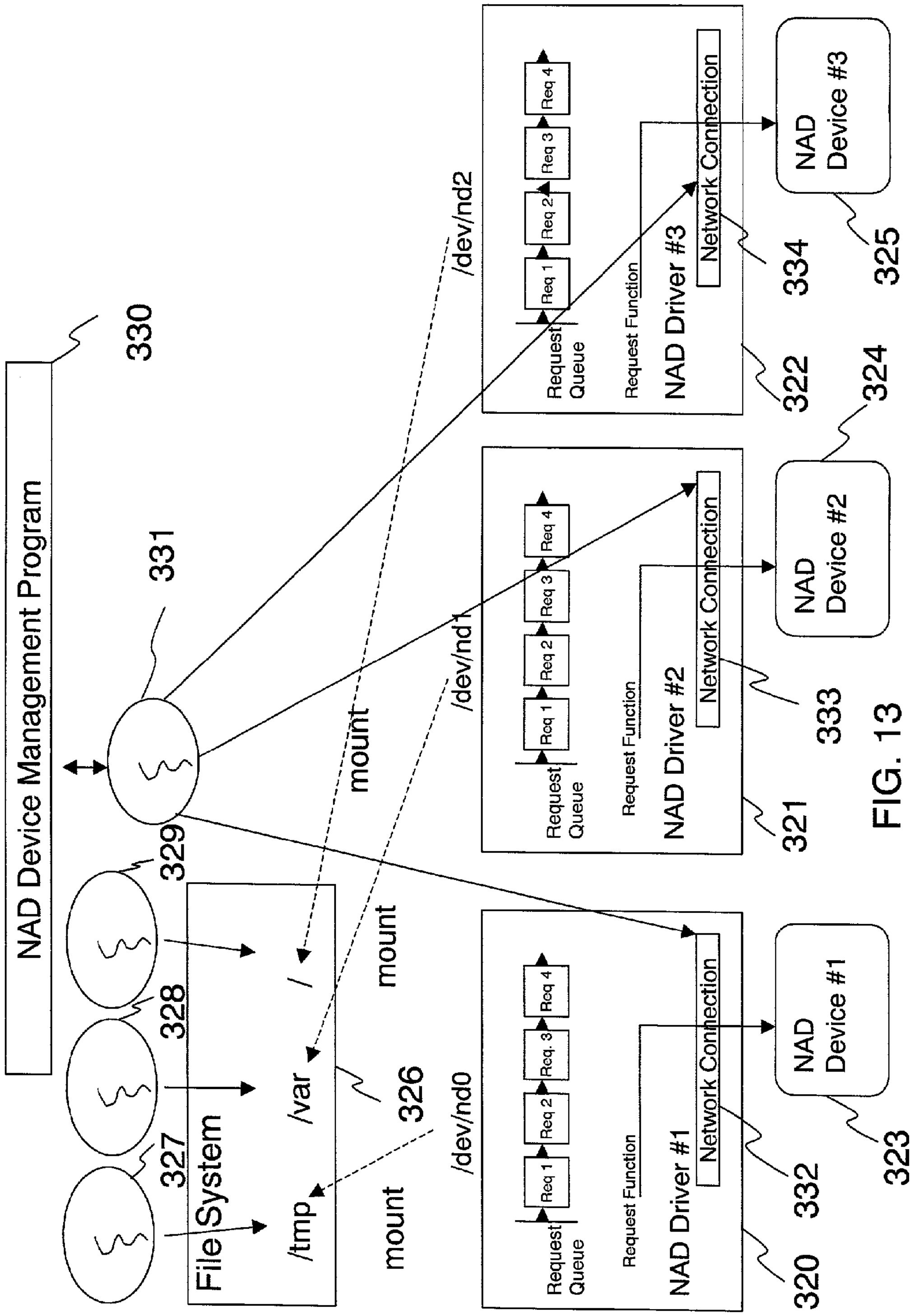
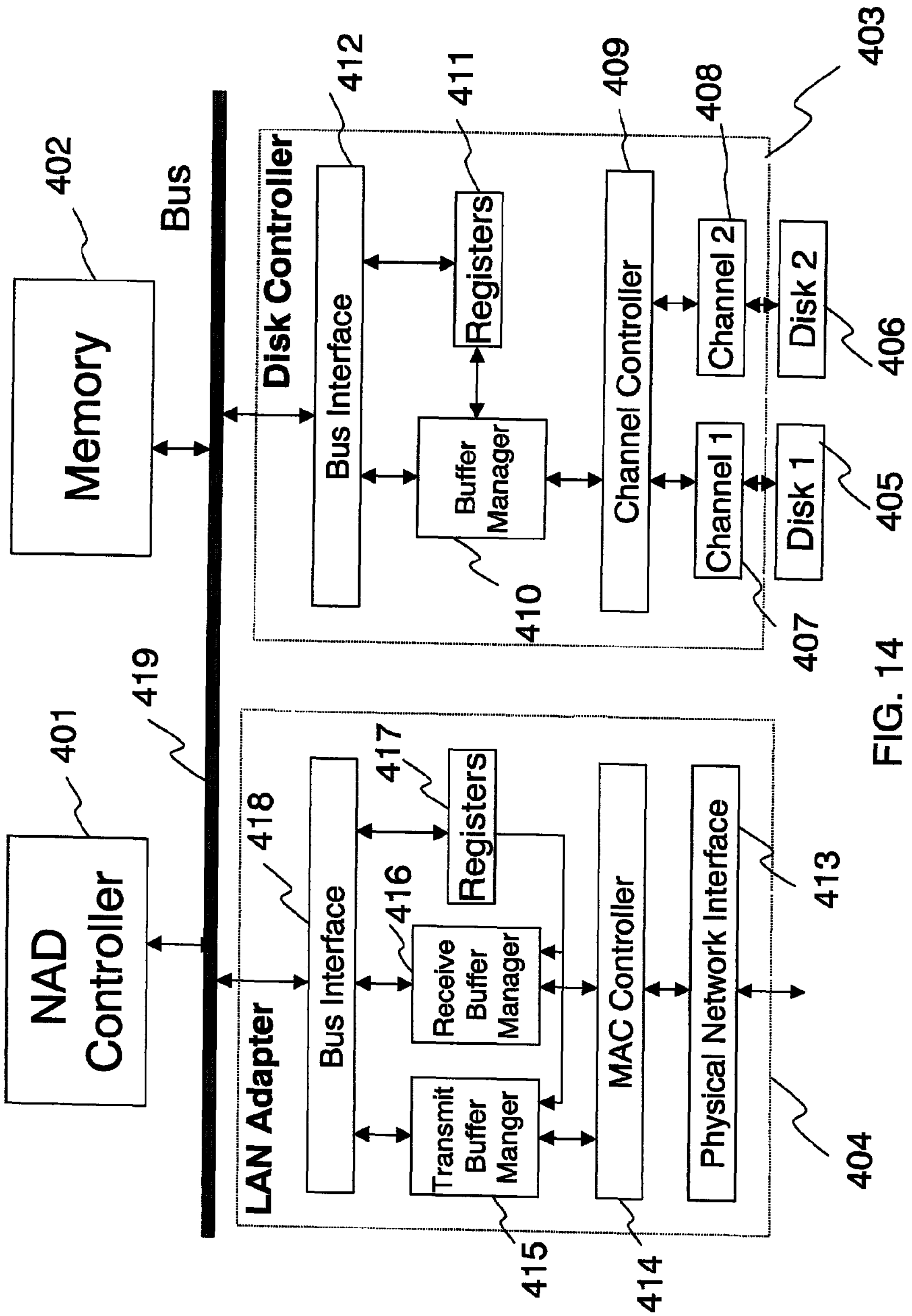

NAD Device Management Program
330
331
327
328
329
File System
/tmp
/var
/
326
mount
mount
mount
/dev/nd0
/dev/nd1
/dev/nd2
Request Queue
Req 1
Req 2
Req. 3
Req 3
Req 4
Request Function
NAD Driver #1
NAD Driver #2
NAD Driver #3
Network Connection
NAD Device #1
NAD Device #2
NAD Device #3
320
321
322
323
324
325
332
333
334

FIG. 13



Memory
402
Bus
NAD Controller
401
419
Disk Controller
403
Bus Interface
412
Registers
411
Buffer Manager
410
Channel Controller
409
Channel 2
408
Channel 1
407
Disk 2
406
Disk 1
405
LAN Adapter
404
Bus Interface
418
Registers
417
Receive Buffer Manager
416
Transmit Buffer Manger
415
MAC Controller
414
Physical Network Interface
413

FIG. 14

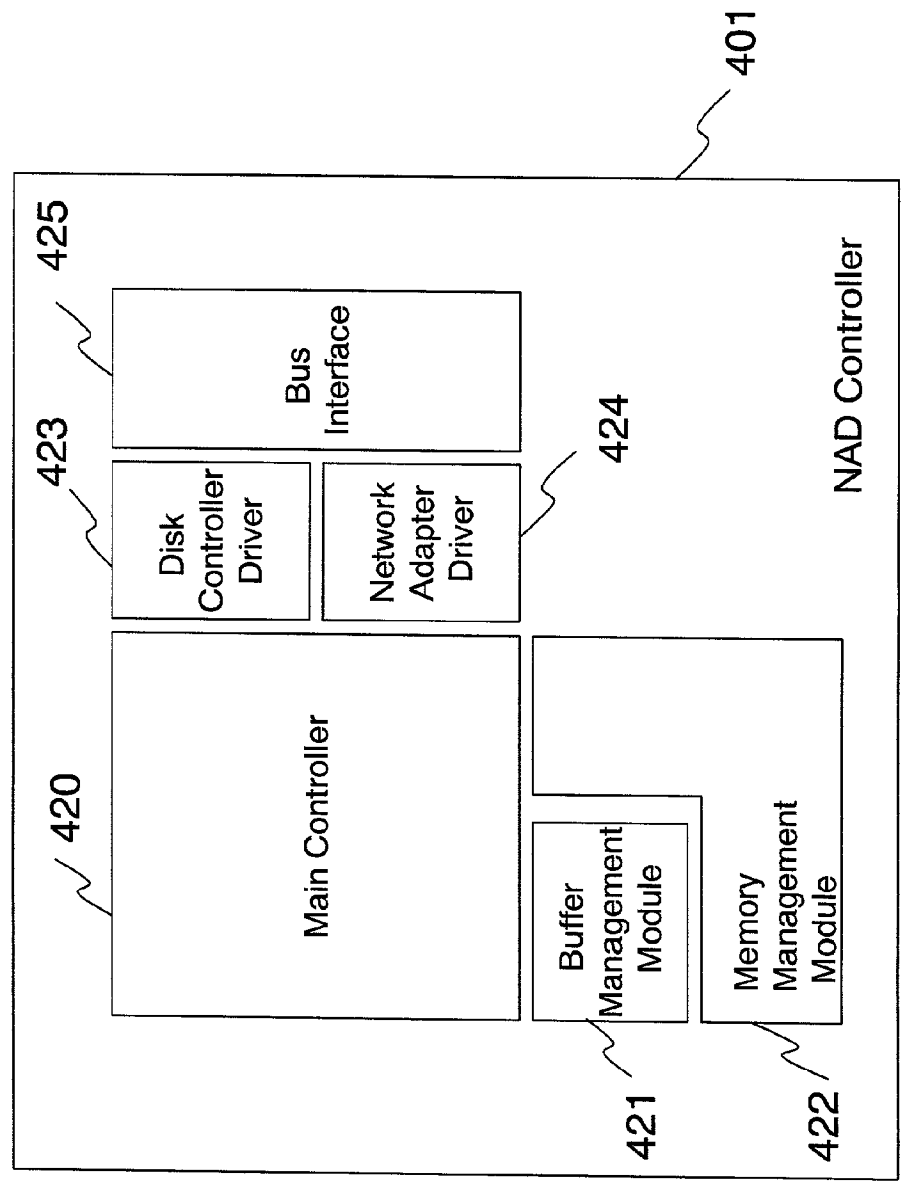
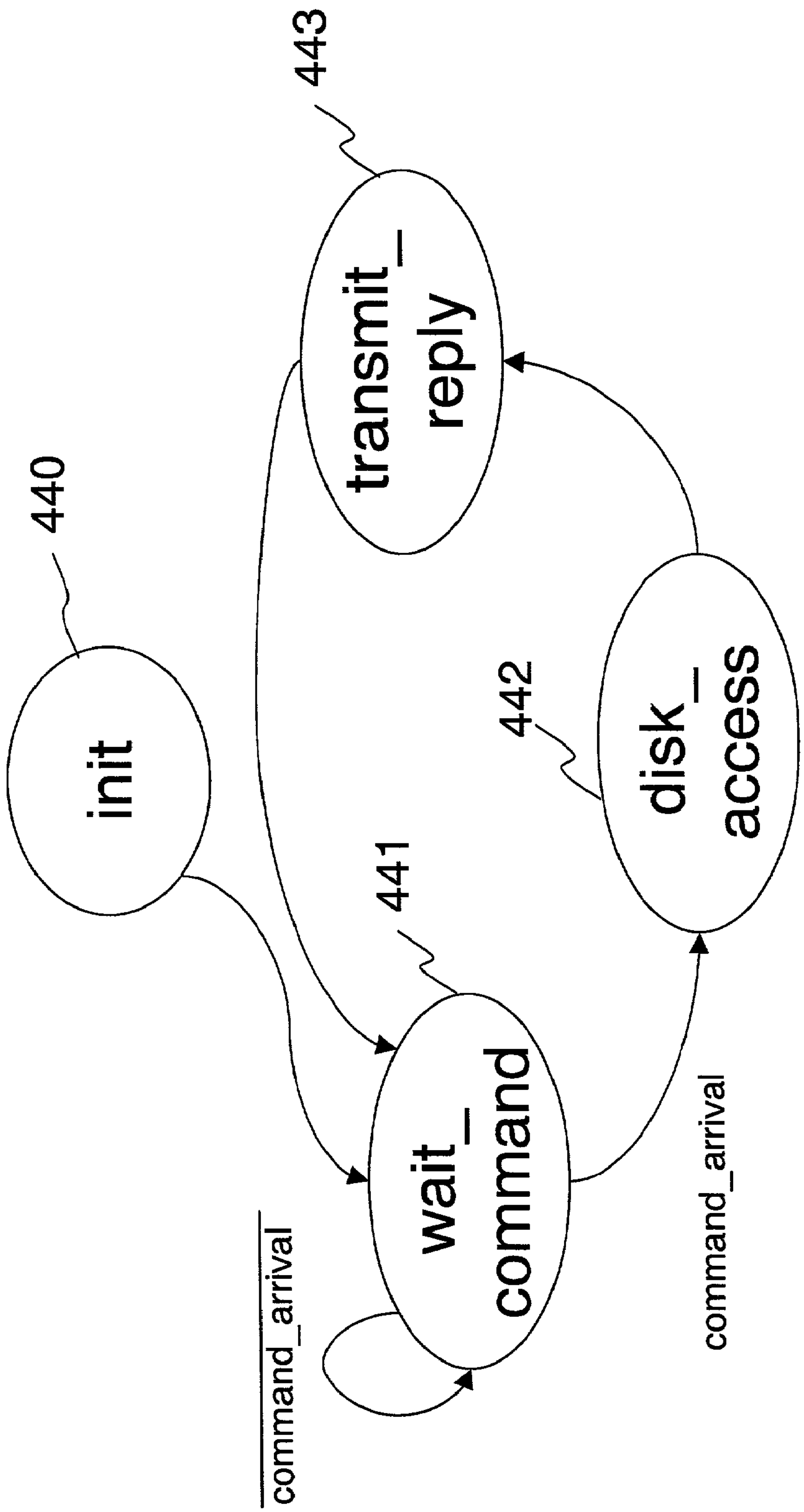

401
NAD Controller
Bus Interface
425
Disk Controller Driver
423
Network Adapter Driver
424
Main Controller
420
Buffer Management Module
421
Memory Management Module
422

FIG. 15



440
init
441
wait_
command
442
disk_
access
443
transmit_
reply
command_arrival
command_arrival

FIG. 16

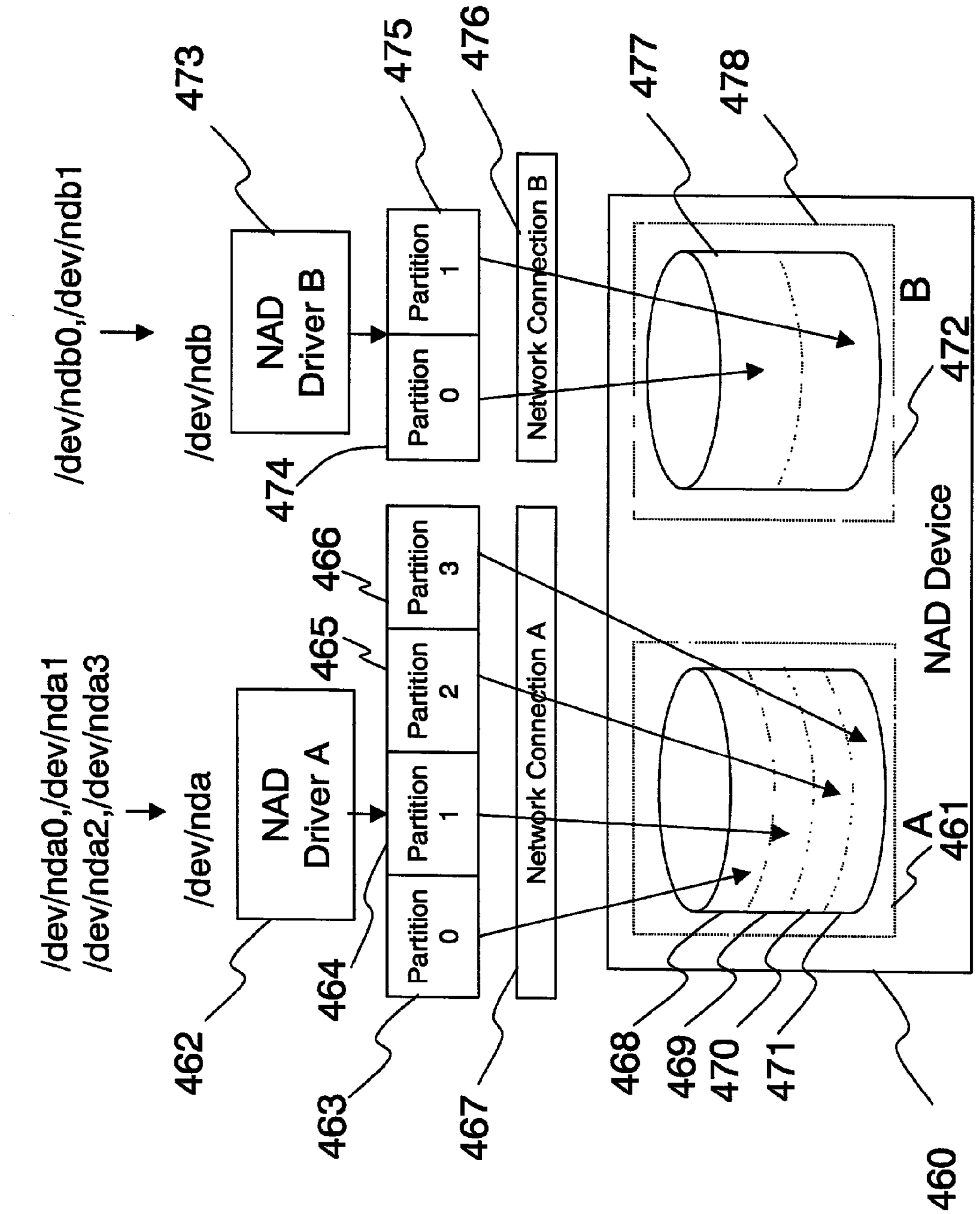

/dev/ndb0,/dev/ndb1
/dev/ndb
NAD Driver B
Partition 0
Partition 1
Network Connection B
/dev/nda0,/dev/nda1
/dev/nda2,/dev/nda3
/dev/nda
NAD Driver A
Partition 0
Partition 1
Partition 2
Partition 3
Network Connection A
A
B
NAD Device
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478

FIG. 17

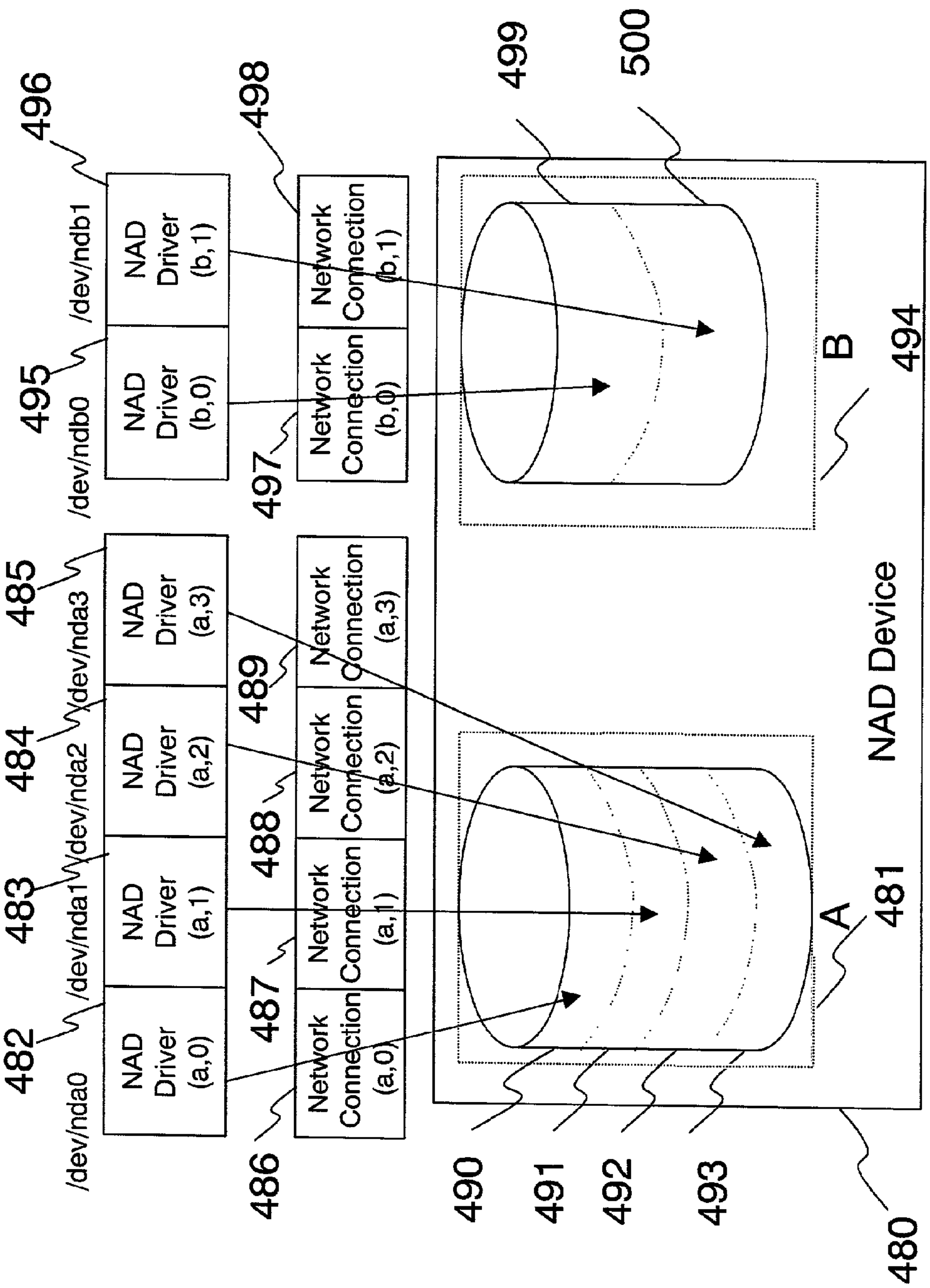

/dev/nda0
/dev/nda1
/dev/nda2
/dev/nda3
/dev/ndb0
/dev/ndb1
NAD Driver (a,0)
NAD Driver (a,1)
NAD Driver (a,2)
NAD Driver (a,3)
NAD Driver (b,0)
NAD Driver (b,1)
Network Connection (a,0)
Network Connection (a,1)
Network Connection (a,2)
Network Connection (a,3)
Network Connection (b,0)
Network Connection (b,1)
A
B
NAD Device
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500

FIG. 18

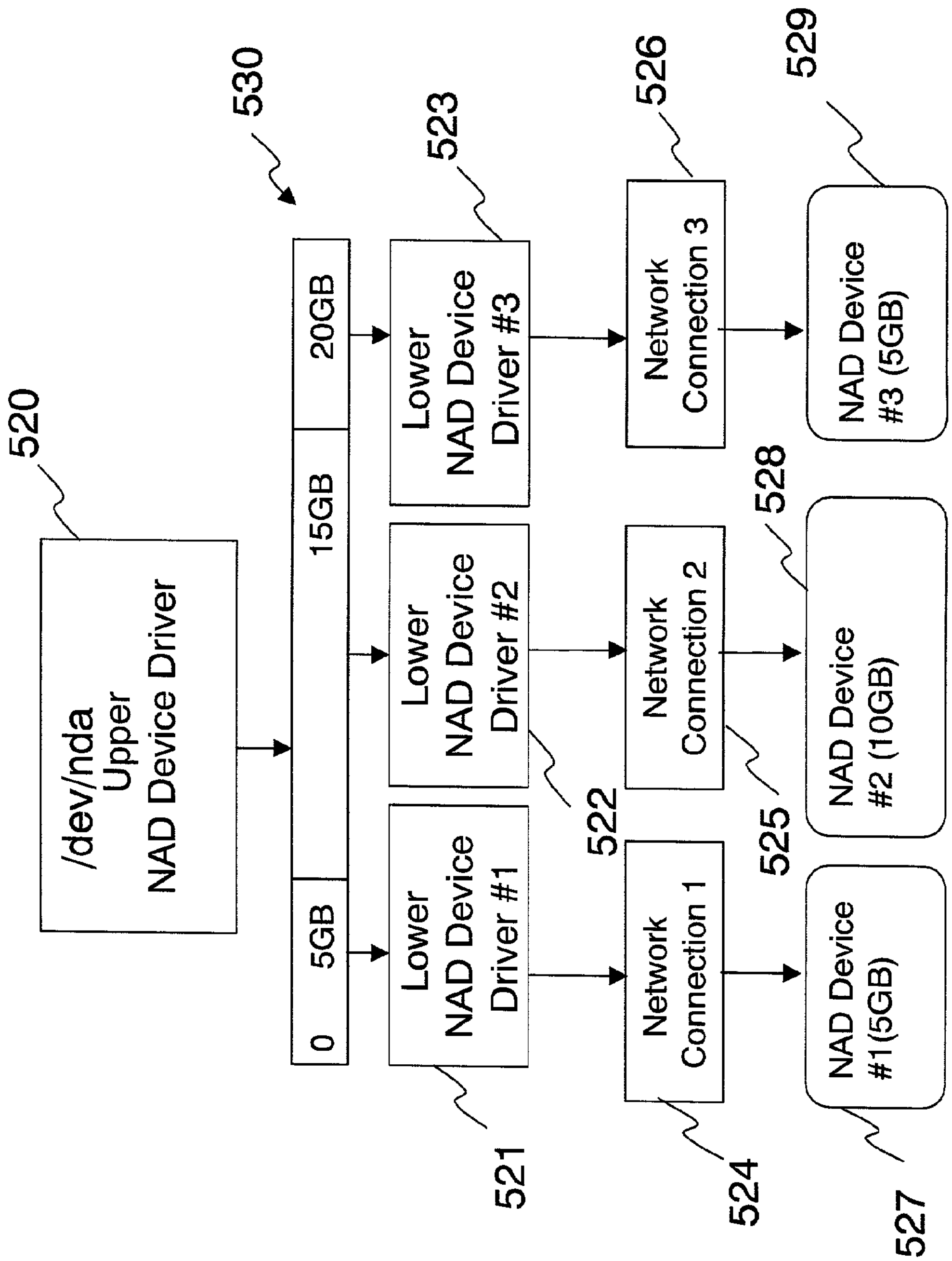

520
/dev/nda
Upper
NAD Device Driver
530
0
5GB
15GB
20GB
Lower
NAD Device
Driver #1
Lower
NAD Device
Driver #2
Lower
NAD Device
Driver #3
521
522
523
Network
Connection 1
Network
Connection 2
Network
Connection 3
524
525
526
NAD Device
#1(5GB)
NAD Device
#2 (10GB)
NAD Device
#3 (5GB)
527
528
529

FIG. 19

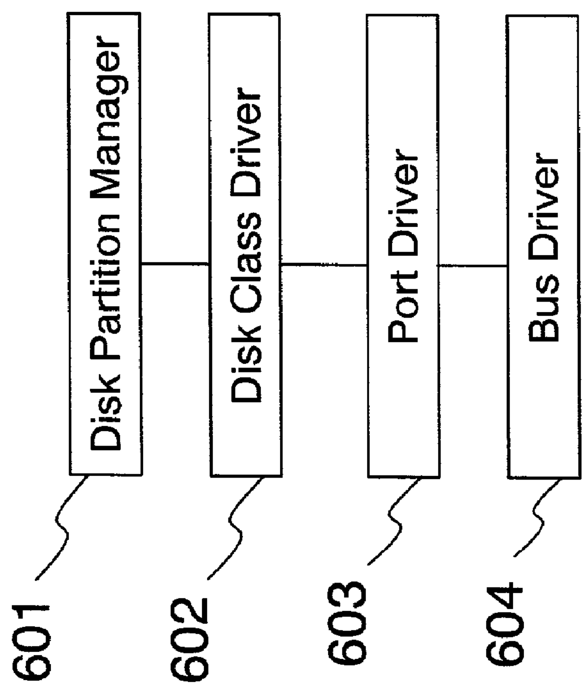


FIG. 20A

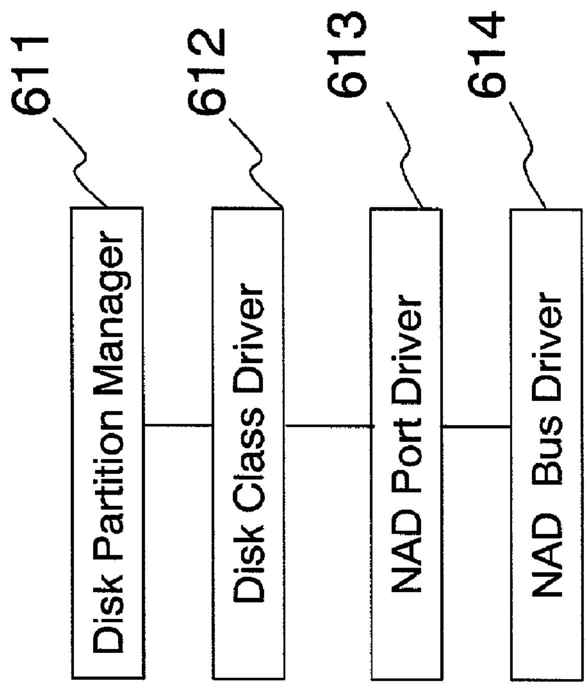
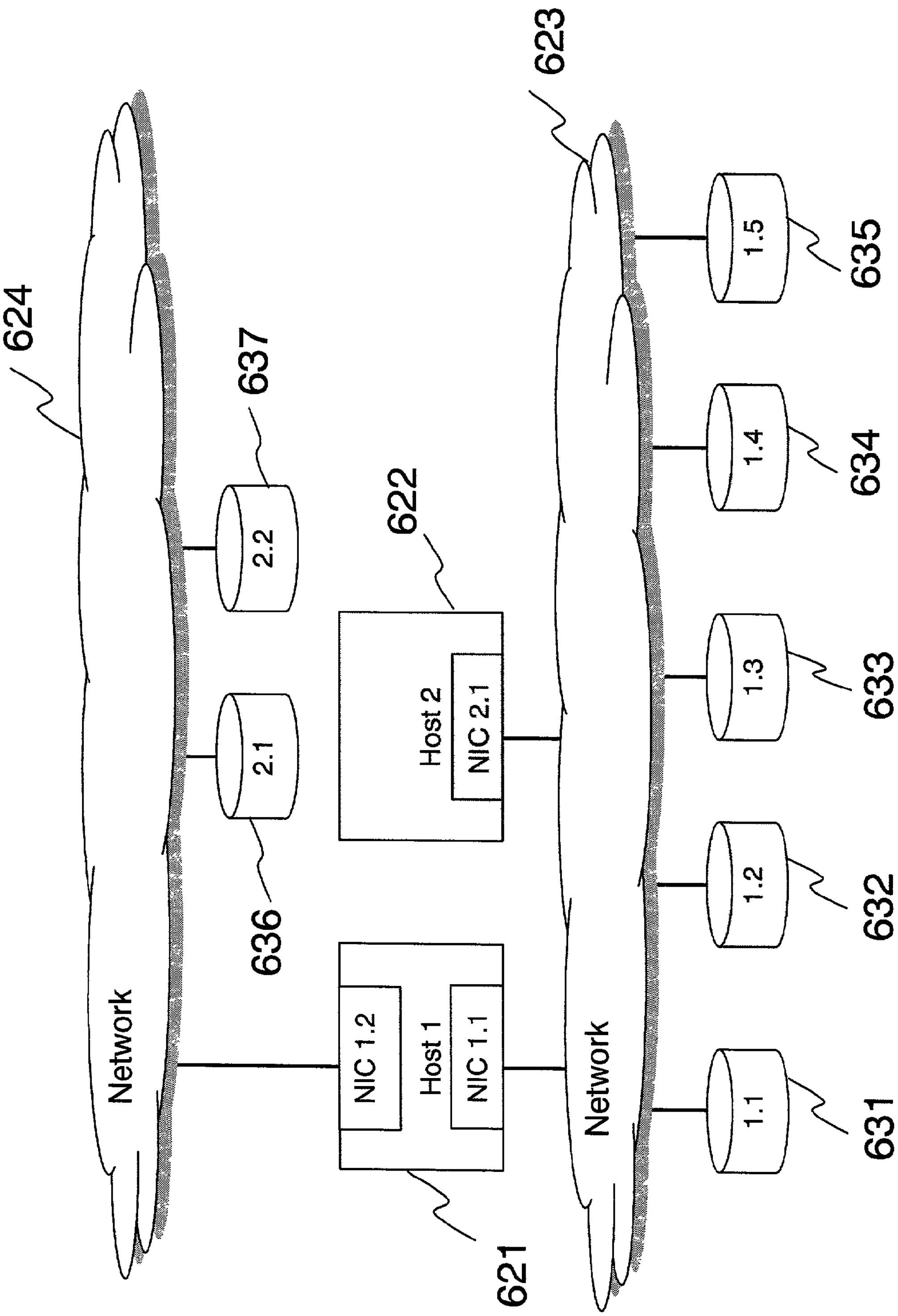
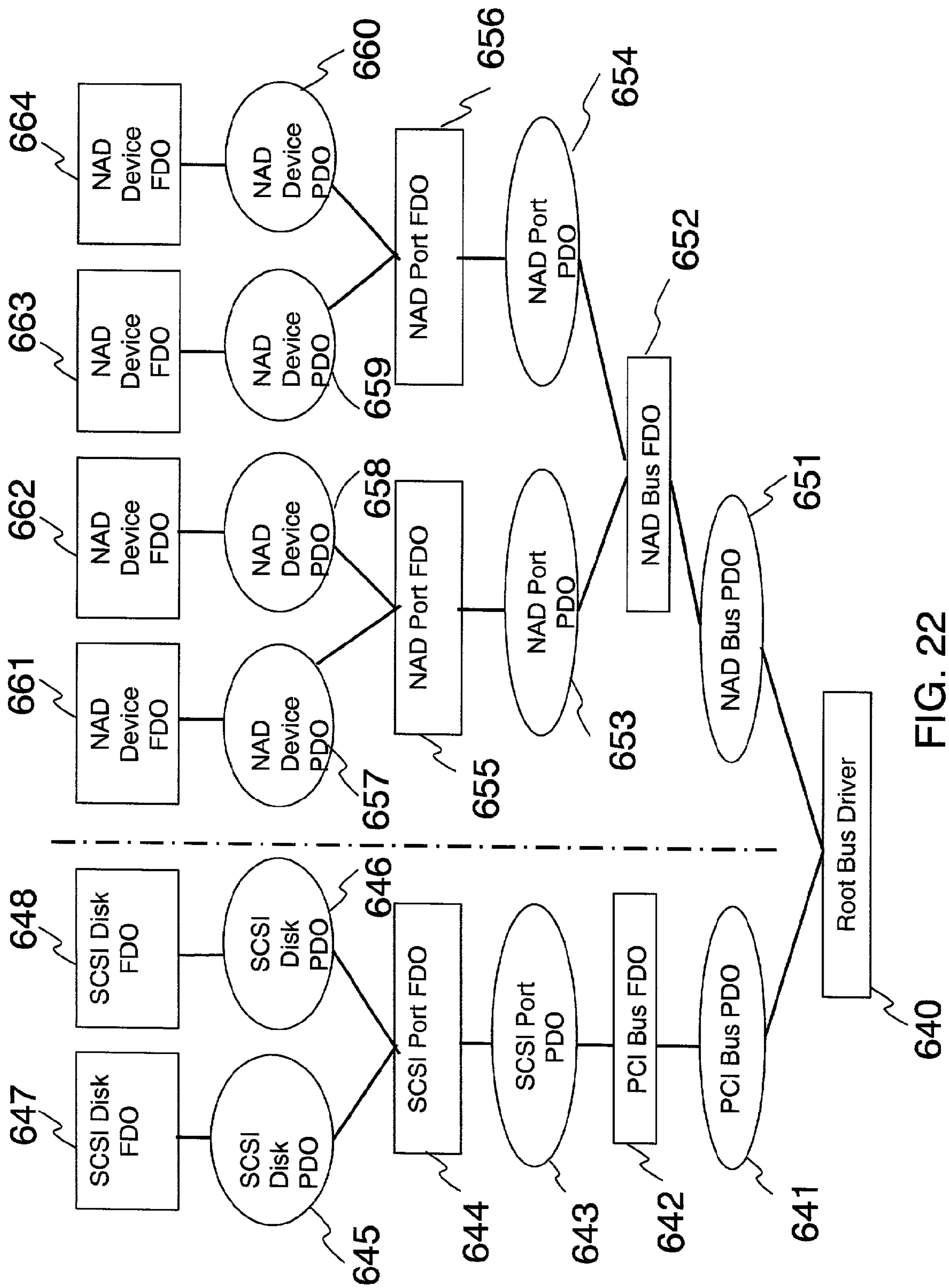


FIG. 20B



624
623
637
622
635
634
633
632
631
636
621
Network
Network
Host 1
NIC 1.1
NIC 1.2
Host 2
NIC 2.1
2.1
2.2
1.1
1.2
1.3
1.4
1.5

FIG. 21



NAD Device FDO
664
NAD Device FDO
663
NAD Device FDO
662
NAD Device FDO
661
SCSI Disk FDO
648
SCSI Disk FDO
647
NAD Device PDO
660
NAD Device PDO
659
NAD Device PDO
658
NAD Device PDO
657
SCSI Disk PDO
646
SCSI Disk PDO
645
NAD Port FDO
656
NAD Port FDO
655
SCSI Port FDO
644
NAD Port PDO
654
NAD Port PDO
653
SCSI Port PDO
643
NAD Bus FDO
652
PCI Bus FDO
642
NAD Bus PDO
651
PCI Bus PDO
641
Root Bus Driver
640

FIG. 22

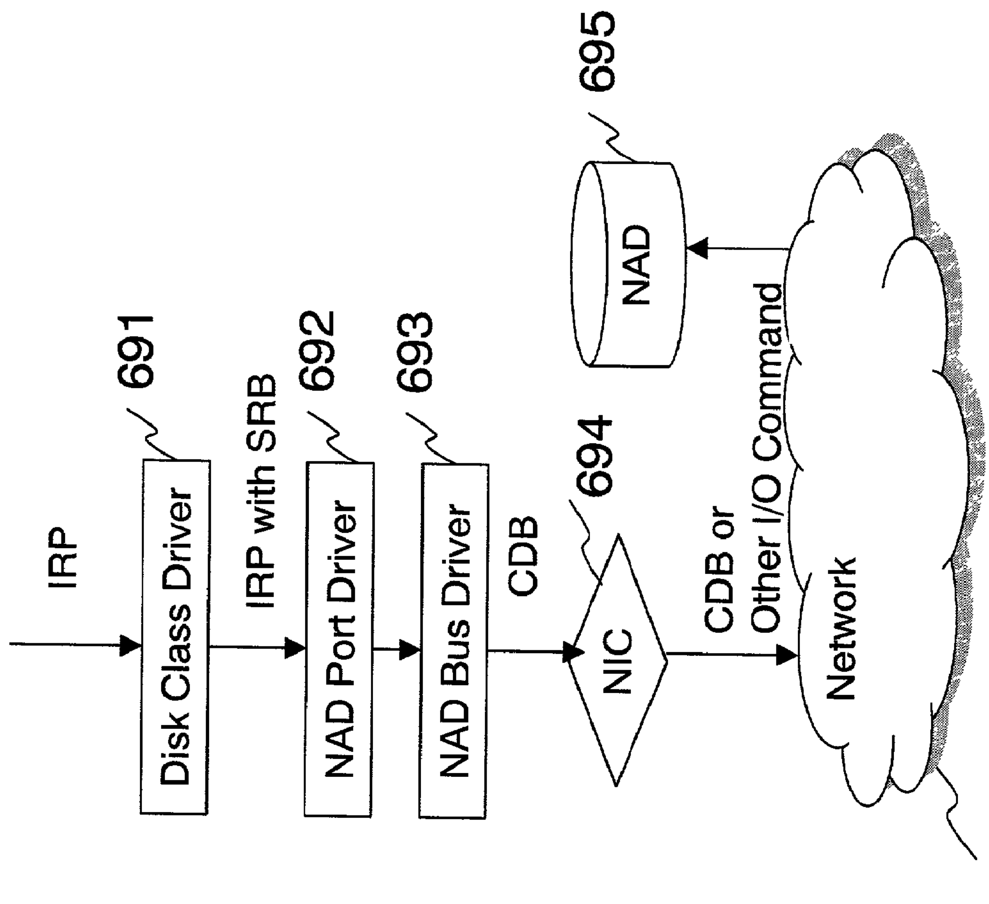


FIG. 23B

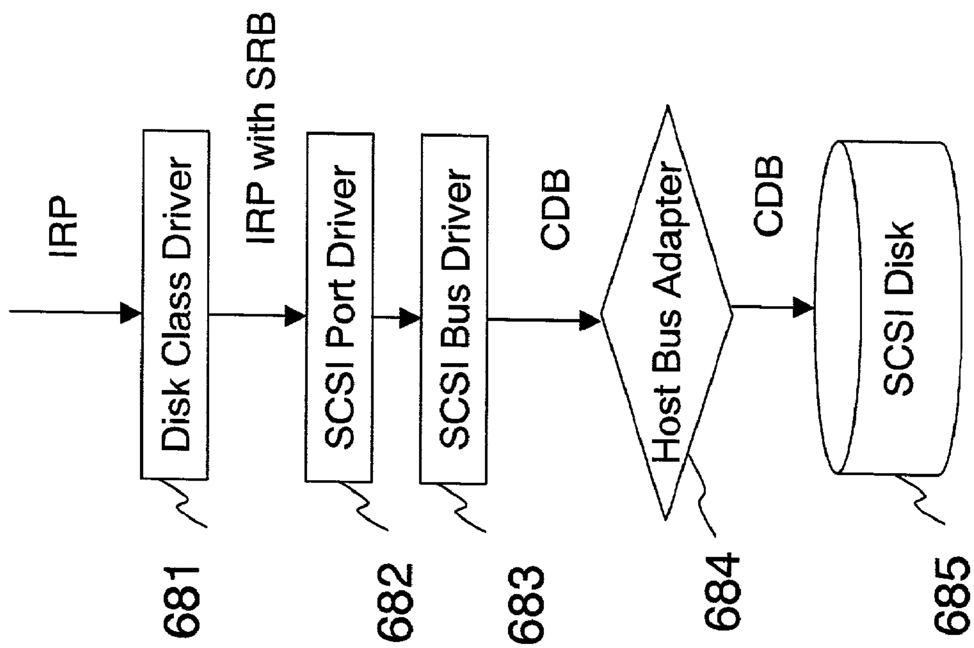


FIG. 23A

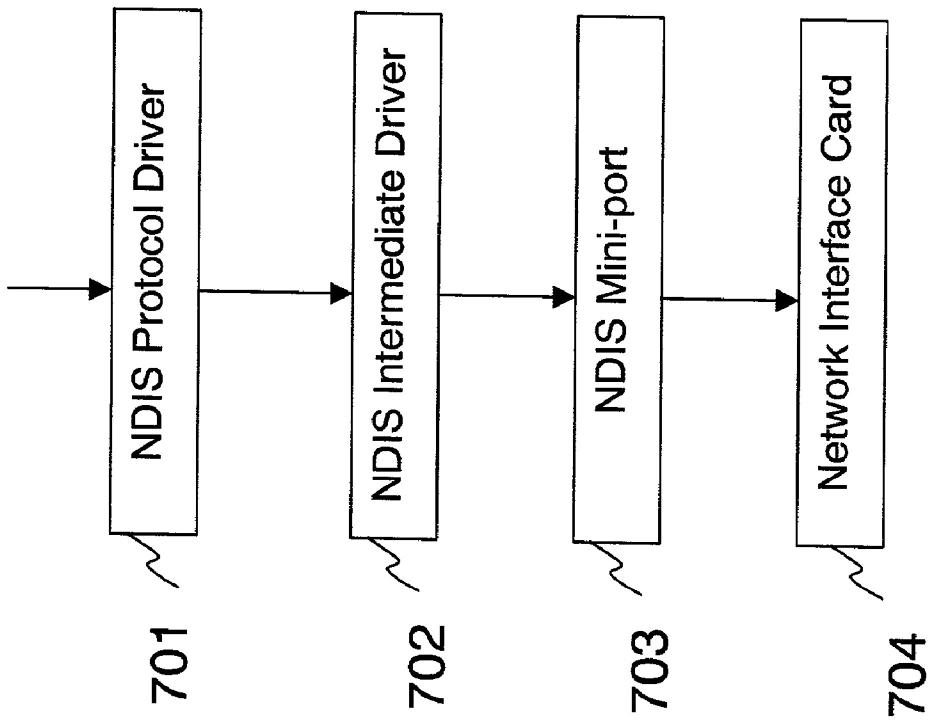

NDIS Protocol Driver
NDIS Intermediate Driver
NDIS Mini-port
Network Interface Card
701
702
703
704

FIG. 24

DISK SYSTEM ADAPTED TO BE DIRECTLY ATTACHED TO NETWORK

CROSS REFERENCE TO RELATED APPLICATIONS

This application is a divisional of U.S. patent application Ser. No. 09/974,082, filed on Oct. 9, 2001 now U.S. Pat. No. 7,792,923, and claims the benefit of U.S. Provisional Application Ser. No. 60/240,344, filed Oct. 13, 2000, the entire disclosures of which are incorporated herein by reference.

BACKGROUND OF THE INVENTION

1. Technical Field

This invention in general relates to computer systems. More specifically, this invention relates to a disk system or interface that can be directly attached to a network.

2. Description of the Related Art

As the Internet becomes popular, the amount of data that needs to be stored has drastically increased. Especially, there is a high demand for a high-capacity disk drive to store multimedia data. For example, a demand for a disk system having a disk capacity of terabytes per server is not unusual.

A tape drive or a CD drive may be used to store such amount of data, but its performance and user convenience are not matched to those of a hard disk drive. However, increasing the capacity of a hard disk in a conventional server system presents some problems.

There are NAS (Network Attached Storage) products that can be connected to a network, usually Ethernet, to provide a pre-configured disk capacity along with integrated system/storage management using the NFS (Network File System) protocol, the CIFS (Common Internet File System) protocol, or both on top of the IP protocol used on the Internet. The primary purpose of these protocols is to exchange files between independently operating computers. Therefore, the client using the NAS for file access experiences the difference between its local storage and the storage in the NAS systems.

The NAS is basically a stripped-down version of a file server having mainly the functions of storing and retrieving files. Accordingly, increasing a disk capacity using a NAS product amounts to adding a separate file server in practice, which presents many shortcomings. Since an NAS disk is not seen as a local disk to the client, the installation, movement, and administration of an NAS disk should be done only through the operating system and software offered as part of the NAS system. An NAS disk is installed in the inside bus of the NAS system, leading to a limitation to the number of disks that can be installed. Since the NAS system has a hard disk under its own operating system, the client cannot use an arbitrary file system to access the hard disk. Further, the NAS system requires an IP address. Overall, not only the installation and administration costs per disk are more expensive than those of a local disk, but also user convenience is severely restricted.

There is SAN (Storage Area Network) that uses the Fibre Channel technology. To use the devices connected to a SAN, a special-type of switch is needed. For example, Fiber Channel uses a Fibre Channel hub or a Fibre Channel switch. The SAN has some shortcomings. Typically, a separate file server is used. In general, the SAN equipment is expensive, and so is the administration cost of the SAN system because, for example, it often requires an administrator with a specialized knowledge on the system.

Therefore, there is a need for an interface that allows a disk system to be directly attached to a network, while still being accessed like a local disk without the need of adding an additional file server or special equipment.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a disk system that can be directly attached to a network connecting to a host without going through a network file system.

Another object is to provide a disk system that can be recognized and used as a local disk to a host without requiring additional facility such as an additional file server, a special switch, or even an IP address, if appropriate.

Still another object of the present invention is to provide a disk system that can be conveniently connected to a server without much intervention of network/server administration.

Yet another object is to provide a low-cost disk system, many of which can be plugged into existing network ports to readily satisfy a disk capacity demand.

Further object is to provide an interface that allows a device attachable to a bus to be plugged into a network port.

The foregoing and other objects are accomplished by providing a network-attached disk (NAD) system that includes an NAD device for receiving a disk access command from a host through a network, a device driver at the host for controlling the NAD device through the network, where the device driver recognizes the NAD device as a local device. The host may run the UNIX or Windows family of operating systems. The NAD device includes a disk for storing data, a disk controller for controlling the disk, and a network adapter for receiving a disk access command from the host through a network port.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of an environment where a network-attached disk (NAD) system of the present invention is used.

FIG. **2** is an illustration of how multiple NAD devices may be accessed by multiple hosts through a network.

FIG. **3** is an example of how multiple NAD devices are accessed by multiple hosts.

FIG. **4** is another example of how each disk inside an NAD may be treated as a separate disk.

FIG. **5** is an illustration of how a block device driver, specifically an NAD device driver, is registered and unregistered under the UNIX operating system.

FIG. **6** is an illustration of the relation among the directory, device file, device driver, and device.

FIG. **7** is an illustration of how a request function directly issues a command to a device.

FIG. **8** is an illustration of how a request function activates a device accessing thread.

FIG. **9** is a block diagram of a local disk system and that of an NAD device running under UNIX.

FIG. **10** is an illustration of a device searching thread for identifying the attached NAD devices and for providing the NAD information to the NAD device management program.

FIGS. **11**A and **11**B are examples of network connections made between an NAD device driver and its corresponding NAD device using a connection setting thread.

FIG. **12** is an illustration of a method of implementing an NAD device driver, using a device accessing thread.

FIG. **13** is an illustration of a method of implementing an NAD device driver, without using a device accessing thread.

FIG. **14** is an example of an NAD device construction.

FIG. **15** is a functional block diagram of an NAD controller.

FIG. **16** is a simplified state transition diagram of a state machine used by the main controller of an NAD controller.

FIG. **17** is an illustration of how a disk inside an NAD device may be divided into disk partitions to which a device driver is assigned.

FIG. **18** is an illustration of how separate NAD device drivers may be generated so that a physically single disk can be assessed by different file systems.

FIG. **19** is an illustration of how the NAD system can recognize physically separate, several NAD disks as a logically single disk.

FIGS. **20**A and **20**B are illustrations of the hierarchies of the disk driver layers in the conventional disk system and the NAD system under the Windows 2000 operating system.

FIG. **21** is an illustration of a network environment where the NAD system of the present invention is used in the Windows 2000 operating system.

FIG. **22** is an example of a device stack created in the Windows 2000 operating system.

FIG. **23**A is an illustration of the flow of IRP, SRB, and CDB in a conventional disk system in the Windows 2000 operating system.

FIG. **23**B is an illustration of the flow of IRP, SRB and CDB in an NAD system in the Windows 2000 operating system.

FIG. **24** is an illustration of NDIS (Network Device Interface Specification) in the Windows 2000 operating system.

DETAILED DESCRIPTION OF THE INVENTION

FIG. **1** shows an environment where the present invention is used. A host **100** has a file system **101**, which may contain a local disk device driver **102** that controls a local disk **104** connected to an internal system bus **103**. A local device is defined as a device inside a standard-alone system as opposed to a network device connected to a network. Local devices are directly connected to a system bus often through an adapter called a host bus adapter allowing the host to communicate with the devices without going through any network, whereas network devices are not directly connected to a system bus, rather connected through an interface called a network interface card (NIC) installed on system bus. The local disk **104** may be a conventional IDE (Integrated Drive Electronics) disk or SCSI (Small Computer System Interface) disk.

The file system **101** also contains a network-attached disk (NAD) device driver **105** of the present invention that controls an NAD device **108** connected through a network adapter device driver **106** and a network **107** such as Ethernet. The NAD device **108** of the present invention contains one or more disks **109**. The network **107** is an existing general-purpose network for carrying storage traffic as well as other application traffic. This so called "front-end" network for carrying general-purpose network traffic is distinguished from a "back-end" network dedicated to storage such as that used in the conventional Storage Area Network (SAN) scheme.

The present invention features two main components: one is the NAD device driver **105** at the host and the other is the NAD device **108** attached to the network.

FIG. **2** shows an example of how multiple NAD devices are accessed by multiple hosts through a network. NAD device #**1** **123** with disk(**1**,**1**) **126** and NAD device #**2** **124** with disk(**2**,**1**) **127** and disk(**2**,**2**) **128** are accessed by Host #**1** **120** through a network **122**, while NAD device #**3** **125** with disk (**3**,**1**) **129**, disk(**3**,**2**) **130**, disk(**3**,**3**) **131** is accessed by Host #**2** **121** through the same network **122**.

Each disk appears to the host as if it is a local disk to connected to the system bus of the host so that each disk can be dynamically installed or removed. The present invention achieves this by creating a virtual host bus adapter in purely software means that recognizes an NAD device as if it is connected to the system bus although there is no physical host bus adapter connected the NAD. This is distinguished from the conventional Network Area Storage (NAS) scheme where a NAS device connected through the MC is still recognized as an independent file server connected to a network.

The Open Systems Interface (OSI) model defines 7 layers of protocols: a physical layer for electrical interface definitions, a data link layer for communication using data frames, a network layer for routing packets from one end to another, a transport layer for dividing messages into packets, a session layer for establishing communication session, a presentation layer for data presentation format, and an application layer for application programs. The present invention uses a data link layer protocol to contain storage commands into data link frames. Because the NAD device is not acting as an independent devices to the host, there is no need to use a network address such as IP address.

Since the specific configuration of the hosts and the disk systems can be dynamically changed, user convenience and portability is preserved as in the case of using a local disk. There is virtually no restriction to the number of disk systems that can be attached to the network, thus providing an unlimited disk storage capacity for a host.

FIG. **3** shows another example of how multiple NAD devices are accessed by multiple hosts through a network. NAD device #**1** **143**, NAD device #**2** **145**, and NAD device #**5** **147** are accessed by Host #**1** **140** through a network **142**, while NAD device #**2** **144** and NAD device #**4** **146** are accessed by Host #**2** **141** through the same network **142**.

The disks contained in an NAD may be treated as separate disks so that each of them can be independently accessed by a host. FIG. **4** shows an example of treating each disk inside an NAD device as separate disks. Disk(**1**,**1**) **166** inside NAD device #**1** **163**, disk(**2**,**2**) **168** inside NAD device #**2**, and disk(**3**,**2**) **170** inside NAD device #**2** are accessed by Host #**1** **160** through a network **122** while disk(**2**,**1**) **167** inside NAD device #**2** **164** and disk (**3**,**1**) **169** and disk (**3**,**3**) **171** inside NAD device #**3** **165** are accessed by Host #**2** **161** through the same network **162**. Note that disk (**2**,**1**) **167** and disk (**2**,**2**) **168**, inside NAD device #**2** **164**, are independently accessed by Host #**1** **160** and Host #**2** **161** respectively.

Block Device Driver

An embodiment of the NAD system will be explained with an example running the UNIX family of operating systems although other operating systems such as Windows may also be used.

Each block device for block data storage, such as a disk device, is assigned a major device number to distinguish among different kinds of block devices, and a minor device number to distinguish among same kinds of block devices. In UNIX, each device is accessed through a device file, which provides an interface for accessing the real device. Device files are usually generated in advance, each with a major device number and a minor device number as well as information on a block device driver.

The purpose of the device driver is to handle the requests made by the kernel with respect to a device. The device driver isolates device-specific codes to provide a consistent interface for the kernel. In order to activate the operation of a device driver, a device file and device driver routines must be prepared, after which the functions of the driver routines must be registered so that the operating system such as UNIX can understand their availability. This is usually done by passing the major number assigned to the device and the functions of the driver routines as parameters.

Registration and Unregistration of Block Device Driver

Once a block device driver is registered by passing the device's major device number and the driver functions as parameters, it may be unregistered by passing the major number.

Table 1 lists the functions used to either register or deregister a device driver.

TABLE 1

Functions	Description
Register__blkdev( )	register a driver by taking a major number and driver functions as parameters
Unregister__blkdev( )	unregister a driver by taking a major number

Table 2 lists the general functions used by the local driver and the NDA driver.

TABLE 2

Driver Function	Description
Read( )	used to read data in the device
write( )	used to write data in the device
ioctl( )	used to change a particular value of a structure for a driver or to control input/output with respect to a device
open( )	used to initialize a driver
release( )	used to eliminate a driver
fsync( )	used to reflect the content of buffer cache to the real device driver
check__media__change( )	used to sense a change in the device condition
revalidate( )	used to update device managed by the deriver and device driver itself

Table 3 lists examples of the driver functions specific to the IDE local disk driver and the NAD driver.

Use of Block Device

Once a block device driver is registered and its device file is generated, read/write is done to the device file to access the real device. The device file, however, is not directly called by the user, rather called after being mounted to the file system. Before being mounted, each block device file must be formatted according to a particular file system. Since the NAD device driver of the present invention is prepared in the same way as a conventional local disk driver, the set of I/O commands used to format a conventional local disk can also be used to format a disk in the NAD device. In addition, since NAD devices are controlled in the device driver level, they can be formatted in a required format independent of the file system used.

FIG. **6** shows the relation among the directory, device file, device driver, and device. The left side shows attachment of a conventional local disk system where a device file **201** mounted on a directory **200** is used by a local disk device driver **202** to control a local disk **203**. The right side shows an NAD system of the present invention where a device file **204** mounted on the directory **200** is used by an NAD device driver **205** to control a NAD device **207** through a local area network (LAN) **206** such as Ethernet. The two relations are similar except that the NAD device is accessed through the network.

Structure of Block Device Driver

Each block device driver has an I/O request queue to store the I/O requests to the device. The stored requests may be re-scheduled for the purpose of improving the performance. Besides the I/O request queue, each block device driver needs a request function to process the I/O requests in the queue.

FIG. **7** shows a situation where the request function directly issues a command to a block device. An NAD device driver **220**, using a device **222** and a file system **223**, has a queue **224** that stores I/O requests **225** through **228**. The NAD device driver **220** has a request function **229** that issues a command to the NAD device **221** by taking a currently processed request **225**.

TABLE 3

Driver Function	IDE Local Driver Function	NAD Driver Function
read( )	ide__read( )	netdisk__read( )
write( )	ide__write( )	netdisk__write( )
ioctl( )	ide__ioctl( )	netdisk__ioctl( )
open( )	ide__open( )	netdisk__open( )
Release( )	ide__release( )	netdisk__release( )
fsync( )	ide__fsync( )	netdisk__fsync( )
check__media__change( )	ide__check__media__change( )	netdisk__check__media__change( )
revalidate( )	ide__revalidate( )	netdisk__ide__revalidate( )

FIG. **5** shows an example where a block device driver, specifically an NAD device driver, is registered and unregistered. Initially, an IDE device driver **181** with major device #**3** is created as well as some null device drivers such as major device #**0 180** and major device #n **182**. Major device #**60**, **183**, the NAD device driver that is assigned a major device number of **60**, is registered by using a device registration function of register_blkdev(**60**, fops) **185**. Later the NAD device driver is de-registered into major device #**60 184**, a null device driver, by using a device de-registration function of unregister_blkdev(**60**) **186**. The figure shows that NAD device driver of the present invention is installed in the same way as the existing block device drivers.

FIG. **8** shows a situation where the request function **229**, instead of directly issuing a command, activates a device accessing thread **230** so that the device accessing thread **230** can issue a command based on the information in the request queue. A thread refers to a single use of a program that can handle multiple users.

Constitution of Local Disk System and NAD System

FIG. **9** shows the constitution of a conventional local disk system and that of the NAD system operating under UNIX. Under a file system **260**, a conventional local disk **264** attached to a local bus **263** is accessed by a conventional local disk device driver **261** through a bus interface **262**. Under the same file system **260**, an NAD device **272** with a disk **273**, attached to a network **271**, is accessed by an NAD device driver **265** through a network interface including a network protocol stack **266**, a network adapter device driver **267**, a bus interface **268**, and a network adapter **270**.

Since an NAD device is to be used like a local disk, the conventional local disk system and an NAD system of the present invention share a basic structure. The difference is that since an NAD system must communicate with an NAD device through a network, a protocol stack is added for network communication. The NAD driver delivers an I/O command to an NAD device through a network adapter and receives a response from the NAD device.

When an NAD device is accessed, either DMA (Direct Memory Access) or PIO (Programmed Input/Output) may be used. A conventional disk device driver operates in a DMA mode by issuing a DMA I/O command to a local disk with a starting buffer address and a byte transfer count. The local disk then takes over the data transfer, after which it interrupts the CPU. Similarly, the NAD device driver may be implemented to operate in a DMA mode by having the NAD device driver deliver an I/O command to an NAD device, which then completes the data transfer, after which it interrupts the CPU.

The conventional disk driver operates in a PIO mode by the CPU transferring data directly through data registers of the disk device until a particular data block is processed. Similarly, the NAD driver may be implemented to operate in a PIO mode by having the NAD device driver deliver a command to an NAD device and continue to transmit/receive data until a particular block of data is processed.

The network protocol that can be used in the present invention is not restricted to a particular protocol. It can be any connection-oriented protocol including TCP/IP. A connection-oriented protocol ensures that packets are not lost and packets are received in the order they are transmitted. If TCP/IP is used, an IP address must be used for each NAD device.

Local Disk Driver and Generation of NAD Driver

Once UNIX starts, if hardware scan detects any conventional local disks, their corresponding drivers are generated according to the units of the local disks or according to the units of disk partitions. In a similar fashion, NAD devices are identified during initial hardware scan and their corresponding drivers acting as a virtual host bus adapter must be generated. The drivers may be generated automatically by using a device searching thread that periodically identifies NAD devices attached to the network or manually by a system administrator using an NAD management program.

FIG. **10** shows a device searching thread for identifying the attached NAD devices and for providing the NAD device information to an NAD device management program. A thread **280** is run in a host **290** through a network protocol stack **282** and a network adapter driver **283** to identify NAD devices **285** through **289** together with the size and device file of each NAD to provide the information **281** to NAD device management program. Once informed of NAD device files available, the user then mounts a selected NAD device file to use a particular NAD device as a local disk.

Network Connection Between NAD Device Driver and NAD Device

In a conventional local disk, disk I/O is performed by reading/writing to I/O ports of the disk controller attached to the internal system bus. But the NAD device driver performs I/O to the corresponding NAD device through a network link. Instead of read/write to an I/O port, I/O is performed by read/write to a network connection such as a socket in UNIX. Therefore, a network connection such as a UNIX socket must be set up between the NAD device driver and NAD device.

FIGS. **11**A and **11**B show examples of network connections between an NAD device driver and the corresponding NAD device using a connection setting thread. NAD device #**1 302** is connected to NAD device driver #**1 301** through a network connection #**1 303** created by the ioctl( ) function, **304** while NAD device #**2 307** is connected to NAD device driver #**2 306** through a network connection #**2 308** created by the ioctl( ) function **309**.

Implementation of NAD Driver

FIG. **12** and FIG. **13** show two methods of implementing an NAD device driver, the former with a device accessing thread, and the latter without a device accessing thread.

FIG. **12** shows three NAD drivers **320**, **321** and **322** with the device files of "/dev/nd0", "/dev/nd1", "/dev/nd2" to access NAD device #**1 323**, NAD device #**2 324**, and NAD device #**3 325**, respectively. Each device file is mounted to "/tmp", "/var", "/" directory in the file system **326**, respectively. User threads **327**, **328** and **329** for accessing the file may read/write on the NAD device through the file system **326**. A connection setting thread **331** provides the list of NAD devices available to an NAD device management program **330**. Based on the user's input, the connection setting thread **331** creates network connections **332**, **333** and **334**, as necessary.

Referring to FIG. **12**, when the user thread requests a file through a file system, the file system first checks the buffer cache to find out whether the requested file block is in the buffer. If the block is in the buffer, the user thread refers to the block. But if the block is not in the buffer, data must be read from the NAD device. The user thread puts the request on the request queue, activates an NAD accessing thread **335** (or **336**, **337**) responsible for NAD device control through a request function, and the user thread blocks itself. The user thread blocked is awakened later by the NAD accessing thread, such as **335**, that received the corresponding data.

FIG. **13** is similar to FIG. **12** except that the user thread now directly requests data from the NAD device rather than using an NAD accessing thread. For example, the user thread puts the request on the request queue, activates a software interrupt that will actually handle block data transfer between the NAD device and the host, and the user thread blocks itself. Once the data transfer is done, an interrupt is generated to wake up the blocked thread.

Communication Protocol Between Host and NAD Device

When a host NAD device driver accesses an NAD device for I/O, the position of the first block and the number of blocks are given as parameters of the I/O command. Or, in the case of SCSI, the I/O command may be in the form of a CDB (Command Descriptor Block).

To transfer the CDB or the block transfer information, a reliable communication protocol is necessary. The present invention is not limited to a particular kind of communication protocol as long as a connection-oriented protocol is used including TCP/IP. A connection-oriented protocol means that packets can be retransmitted in the case packets are lost, and received packets are arranged at the receiver end in the order they were sent.

NAD Device

FIG. **14** shows a functional block diagram of the NAD device of the present invention. A preferred embodiment of the NAD device is comprised of an NAD controller **401** for controlling the whole NAD device, memory **402**, a disk controller **403** for executing a disk access command, one or more disks **405**, **406**, and a LAN adapter **403** for receiving a disk access command from a host through a network. The NAD controller **401**, the memory **402**, the disk controller **403**, and the LAN adapter **404** are all connected to a bus **419** internal to the NAD device.

The disk controller **403** is a module that performs disk I/O operations by controlling the disks **405** and **406** over internal disk channels. The disk controller **403** is further comprised of one or more disk channels **407** and **408** controlled by a channel controller **409**, a buffer manager **410**, some registers **411**, and a bus interface **412**. The buffer manager **410** consults the registers **411** to obtain a disk sector number and a channel to execute a disk access command. The buffer manager **410** also commands the channel controller **409** to transfer data from the memory to disk channel **407** or **408** or vice versa as a result of executing a disk access command. The channel controller **409** actually accesses the disk over the disk channel **407**, **408** to transfer data from the disk to the memory or vice versa.

The LAN adapter **404** is a module that receives disk I/O command packets from the host and transmits replay packets over the network. The LAN adapter **404** is further comprised of a physical network interface **413** for interfacing with the network, a MAC (media access control) controller **414**, transmit buffer **415** for storing transmit data, a receive buffer **416** for storing receiving data **416**, registers **417**, and a bus interface **418**.

The bus interface **418** transfers data from the bus to the transmit buffer **415**, the receive buffer **416**, and the registers **417**, or vice versa. The MAC controller **414** transfers data to the physical network interface **413** so that the physical network interface can transmit the data to the host computer. When the physical network interface **413** receives a disk I/O request packet from the host computer, it transfers the packet to the MAC controller **414** so that the MAC controller can extract necessary data from the packet and transfer the data to the receive buffer **416**.

FIG. **15** shows that the NAD controller **401** may be comprised of a main control **420** for controlling the NAD, a buffer management module for caching data in the disk **421**, a memory management module for managing assignment of memory space **422**, a disk controller driver **423** for interfacing with the disk controller, a network adapter driver **424** for interfacing with the network adapter, and a bus interface **425** for interfacing with the bus inside the NAD.

The NAD controller **401** mainly executes I/O commands from the host's NAD device driver, but it can perform other additional functions. For example, a filter program can be installed to NAD so as to provide features that are not offered in the host, for example, a back up operation. Other examples include access control, access share, access right transfer, etc. Specifically, a filter program can be installed to limit access to an NAD device to a certain time period, to allow several hosts simultaneously access an single NAD, or to transfer one host's access rights to another host. The NAD device driver at the host can request to execute the filter program at the time of I/O command execution through the ioctl( ) function in UNIX.

FIG. **16** shows a simplified state transition diagram of a state machine used by the main controller **420**. At the 'init' state **440**, the state machine initializes all the NAD hardware and allocates memory for the disk controller **403** and the LAN adapter **404**. Upon completing the initialization process, the state machine makes a transition to 'wait-command' state **441** where the NAD system waits for an incoming I/O command from the host computer over the network. When such I/O command is received from the host computer, the state becomes the 'disk_access' state **442** where an appropriate disk I/O operation is performed through the disk controller. Upon completing the disk I/O, the state moves to the 'transmit_reply' state **443** where the NAD device sends the result to the host computer through the LAN adapter **404**. A person skilled in the art would appreciate that the state machine can be readily realized with a CPU and memory.

Network Adapter Driver and Disk Controller Driver

The network adapter and the disk control driver can be implemented at least in two ways. One uses an interrupt mechanism through DMA (Direct Memory Access) and the other uses polling through PIO (Programmable I/O). The former has the advantage of easy programming so that other jobs can be executed without a complete disposition of disk controller data. The latter has the advantage of dispensing with time delay due to interrupts, but has the disadvantage of an inefficient processor usage due to the time spent for continuous read and write.

Additional Embodiments of NAD Drivers

Usually, an NAD device driver is generated for each disk unit attached. However, just as a local disk may be partitioned, the disk inside an NAD device may also be partitioned into several disk partitions where each disk partition can be accessed by a separate device driver. Alternatively, several disks located in the physically separate NAD devices may be combined for use as a logically single disk.

FIG. **17** shows an example where the disk inside an NAD device may be divided into several disk partitions where all of the partitions are assigned a single device driver. An NAD driver A **462**, for example, is assigned to four partitions **463**-**466** so that the NAD driver A **462** refers the partition table in order to handle I/O requests directed to specific partitions **468** through **471** of a disk **461** inside a NAD device **460**, respectively, using a same network connection **467**. Similarly, an NAD driver B **473** is assigned to two partitions **474** and **475** so that the NAD driver B **473** can be used to control two disk partitions **477** and **478** of a disk **472** inside the NAD device **460**.

FIG. **18** shows an example where separate NAD device drivers may be generated so that a physically single disk can be assessed by different file systems. Disk A **481** inside an NAD device **480** is divided into four partitions **490** through **493**, and four separate NAD driver (a,**0**) through driver (a,**3**) **482** through **485** are created so that each NAD driver can control each disk partition through a separate network connection **486** (**487**, **488**, or **489**). Similarly, disk B **494** inside the NAD device **480** is divided into two partitions **499** and **500**, and two separate NAD drivers (b,**0**) and NAD drive (b,**1**) **495** and **496** are created so that each NAD driver can control each disk partition through a separate network connection **497** (or **498**). Since different network connections are used, this configuration enables a physically single hard disk to be mounted to different file systems.

FIG. **19** shows an example of how the present invention can recognize physically separate, multiple disks in different NAD devices as a logically single disk. Specifically, FIG. **19** shows that three lower-level NAD device drivers **521**, **522** and **523** controlling NAD device #**1** **527** of 5 GB, NAD device #**2** **528** of 10 GB, and NAD device #**3** **529** of 5 GB, respectively, through separate network connections **524**, **525** and **526**, are united into a single upper-level NAD driver **520** partitioned into a configuration **530**. The file system mounts "/dev/nda" to access the total space of 20 GB.

NAD System Running Under Windows Operating System

The foregoing system and method explained using examples running under the UNIX family of operating systems can equally be applied to implement an NAD system running under the Windows™ family of operating systems so that it can be recognized as a local disk. For example, an NAD device may treated as a local disk per se by a Windows 2000™ host so that all disk operations exercised by the host control a local disk, including formatting and partitioning, can be done to the NAD device.

This feature differentiates the present invention from other solutions, such as those provided by the NAS technology, which expand disk space through the intervention of a file system instead of directly adding individual disks at the device level of the host system. At the same time, since the NAD device is to be accessed through the network, the present invention redirects the disk I/O request to the network interface otherwise would be directed to the disk controller connected to the inside system bus in the case of using conventional local disks.

In other words, the present invention creates a virtual host bus adapter in purely software means by modifying a driver at the host so that the host recognizes the NAD device as if it is connected to the system bus through a physical host adapter although actually there is no physical host adapter connected to the bus. Since an NAD device is accessed as if it is a local device connected to the internal bus of a host, there is no need to use network addresses such as IP addresses for the host to communicate with the NAD device. Instead, data link frames containing storage commands are exchanged between the host and the NAD device.

FIGS. **20**A and **20**B shows a comparison of the hierarchy of the disk driver between the conventional disk system and the NAD system of the present invention. FIG. **20**A shows conventional disk driver layers in Windows 2000, which are organized in a hierarchy comprising a disk partition manager **601**, a disk class driver **602**, a port driver **603**, and a bus driver **604**.

In the Windows 2000 operating system, the generic term, 'bus', refers to a piece of hardware to which devices connect electronically. Not only does it include things like the PCI bus, but it also includes anything that can have multiple devices plugged into it such as a SCSI adapter, a parallel port, a serial port, a USB hub, and so on. One responsibility of the bus driver is to enumerate devices attached to the bus and to create physical device objects for each of them as necessary in Windows 2000. Therefore, the bus driver is a collection of software routines that contain the information about the specific bus and the functions that allocate system resources such as port addresses and IRQ numbers to the devices connected to the bus. It is the port driver that contains routines required to perform most of the actual disk I/O operations.

The major feature of the present invention is to replace the conventional bus driver and the port driver with a new bus driver and a new port driver so that the NAD devices can be recognized as the same as the local disks and the disk I/O operations can be performed to the NAD devices through the network port of the Windows 2000 host.

FIG. **20**B shows the driver layers of the present invention, which have an NAD port driver **613** and an NAD bus driver **614** replacing the corresponding conventional Windows 2000 driver layers of FIG. **20**A. The NAD bus driver **614** implements a virtual host bus adapter, through which disk I/O operations are to be done to and from a set of NAD devices. The NAD port driver **613** implements a set of routines required to perform actual disk I/O operations by redirecting the I/O requests to the NAD devices through the network port of a Windows 2000 host.

FIG. **21** shows an example of a network environment where NAD devices of the present invention are attached to multiple hosts. The example shows that both Host #**1** **621** and Host #**2** **622** run Windows 2000 connected to Network #**1** **623** and Network #**2** **624**. Host #**1** uses disk(**1**,**1**) **631** and disk(**1**,**3**) **633** through Network #**1**, disk (**2**,**1**) **636**, and disk(**2**,**2**) **637** through Network #**2** **625**. Similarly, Host #**2** uses disk(**1**,**2**) **632**, disk(**1**,**4**) **634** and disk(**1**,**5**) **635**.

Given the NAD bus driver and the NAD port driver, a Windows 2000 system creates device stacks as specified in Windows 2000 in order to be able to process I/O requests. Each device in Windows 2000 is expressed in terms of device objects organized in a stack structure. Device objects are data structures that the Windows 2000 system creates to help software manage hardware. Many of these data structures can exist for a single piece of physical hardware. The lowest-level device object in a stack is called a physical device object (PDO). Above a PDO in a device object stack is an object called a functional device object (FDO). There may be a collection of filter device objects below and above the FDO. The Plug and Play (PnP) Manager component of Windows 2000 constructs the stack of device objects at the command of device drivers. The various drivers that occupy the stack for a single piece of hardware perform different roles. The function driver manages the device, and the bus driver manages the connection between the device and the computer.

FIG. **22** shows an example of device stacks that may be created to implement the present invention, where all filter device objects are omitted for the simplicity. Shown on the left half is a layer of recursively enumerated SCSI devices on top of the PCI bus, which is typically the case when SCSI disks are connected to the host's PCI bus. In the first instance, a PnP Manager has a built-in driver for a virtual root bus that conceptually connects computer to all the hardware that can't electronically announce its presence including hardware bus such as PCI. The root bus driver **640** gets information about the PCI bus from the registry to create a PCI bus PDO **641**, a PDO for the PCI bus, where the registry was initialized by a Windows 2000 setup program.

Having created the PCI bus PDO **641**, the PnP Manager then loads functional drivers for the PCI bus, thus creating a PCI bus FDO **642**. The functional driver of the PCI bus can then enumerate its own hardware devices attached to the PCI bus, where the example system in FIG. **21** assumes to have a set of SCSI devices, to create a SCSI port PDO **643**. Once the SCSI port PDO **643** is created, the PnP Manager then loads drivers for SCSI port device, thus creating a SCSI port FDO **644**. Similarly, SCSI disk PDOs, such as **645** and **646**, are created for each of the individual SCSI disks on top of the SCSI port, and SCSI disk FDOs, such as **647** and **648**, are in turn created by loading the disk class driver.

Shown on the right half of FIG. **21** is the corresponding device stacks for the NAD devices that would be created by using the NAD bus driver and NAD port driver replacing the PCI bus driver and the SCSI port driver, respectively. On top of the root is a NAD BUS PDO **651**, the PDO of the NAD bus that is not conventional hardware bus such as PCI, but a bus required to fit in the Windows 2000 device stack in order to provide virtual bus for NAD devices. On top of the NAD bus PDO **651**, the PnP Manager creates a NAD bus FDO **652** by loading a NAD bus driver.

A set of NAD Port PDOs **653** and **654** for each of individual network interface cards (NICs) of the Windows 2000 host are then created recursively since one NAD port is implemented to correspond to one NIC of the host in the present invention. On top of each NAD port PDO such as **653** or **654**, each NAD port FDO such as **655** or **656** is created by loading a NAD port driver. It is the NAD port driver that performs the actual NAD disk I/O operations. The NAD port driver should handle the NAD device I/O requests by redirecting the I/O requests and obtaining the I/O replies to and from the corresponding NAD devices through the specific NIC. The NAD port FDO such as **655** or **656** then creates individual NAD device PDOs such as **657**, **658**, **659** or **660** on top of the specific NAD port for individual NAD devices that can be accessed through the specific NAD port bound to a specific NIC.

FIG. **22** shows that for the example in FIG. **21**, two stacks of NAD port objects **653** and **654** are created because Host #**1** has two NICs. Host #**1** also has four NAD device PDOs **657** through **660**, two for each NAD port, because NAD devices, i.e., disk(**1**,**1**) **631** and disk(**1**,**3**) **633** and disk(**2**,**1**) **636** and disk(**2**,**2**) **637** are to be accessed through the NIC(**1**,**1**) and NIC(**1**,**2**) respectively. For each individual NAD device PDO such as **657**, **758**, **659** or **660**, the PnP Manager loads disk class driver to create a NAD device FDO such as **661**, **662**, **663** or **664**.

Note here that the only NAD bus driver and NAD port driver are to replace the conventional bus driver and SCSI port driver respectively in order to substitute the NAD devices for the conventional local disks. Disk class driver and other higher level drivers of Windows 2000 should remain intact without a single change in order for the Windows 2000 system to recognize the NAD device as same as a local disk.

In Windows 2000, each request for an operation affecting a device uses an I/O request packet (IRP). IRPs are normally sent to the topmost driver of a stack for the device and can percolate down the stack to the other drivers. At each level, the driver decides what to do with the IRP. Sometimes, the driver does nothing except passing the IRP down. The driver may completely handles the IRP without passing it down or process the IRP and pass it down. In the case of disk I/O, for example as shown in FIG. **20**B, an IRP for a file I/O sent to the file system driver is passed to a volume manager, a disk class partition manager, to a partition manager, and to disk class driver.

It is the disk class driver where a SCSI Request Block (SRB) is created to be included in the IRP as necessary. An SRB is a data structure specified in the Windows 2000 for SCSI device I/O. If the IRP is for the conventional local disk, the disk class driver passes the new IRP down to a SCSI port driver that completes actual disk I/O operation. If the IRP is for the NAD device connected to the network, the disk class driver passes the IRP down to NAD port driver that completes NAD device I/O through the network interface.

Without regard to the particular device type of the disk, local or NAD device, it is the feature of the Windows 2000 device stack as shown in FIGS. **20**A and **20**B that an IRP for a specific disk, local or network-attached, is directed eventually to the corresponding disk. This is because separate disk object stacks are created for each of the individual disks. FIG. **22** shows that separate SCSI disk FDO/PDOs and NAD device FDO/PDOs are bound to each of the individual local disks and NAD devices, respectively.

The present invention replaces the conventional disk bus driver and port driver with the new NAD bus and port drivers as shown in FIG. **20**B so that NAD devices would be recognized as local disks by the Windows 2000 system.

All of the Windows 2000 device drivers have functions to create and remove the FDO for each device and dispatch functions to handle IRP passed down from the above driver layer. The major and minor function numbers in the IRP specify which of the dispatch functions will be invoked.

The following is an explanation of the actual software modules implemented in the NAD bus driver and port driver of the present invention in order to implement the NAD system for Windows 2000.

NAD Bus Driver

The NAD bus driver is a set of software modules that implement a virtual host bus adapter to which NAD ports are to be attached, where the individual NICs of a host are realized as NAD ports. The functions of the NAD bus driver are basically the same as those of a conventional PCI bus driver in Windows 2000. The NAD bus driver performs the functions of finding out the number of the NICs installed in the host computer and enumerating those NICs to create an NAD port PDO for each of the existing NICs. It also performs the functions of creating, starting, stopping, and removing an NAD port. In the NAD system, an individual NIC is regarded as an independent NAD port so that NAD disk ports for NAD devices are created according to the number of independently operating network units. See the example configuration shown FIG. **22**.

The difference between the NAD bus driver and a conventional PCI bus driver is that the NAD bus driver is for NAD devices that are physically separated from the system bus of the host but are connected through network ports. Unlike a conventional Windows 2000 system that detects plug-in of a device to or removal of a device from the hardware bus through a hardware interrupt, the NAD bus driver is implemented by creating a kernel thread to install and remove an NAD port on the NAD bus. The kernel thread created by the NAD bus driver starts to work when an IRP with IRP_MJ_PNP as its major function number and IRP_MN_START_DEVICE as its minor function number is sent to the NAD bus FDO from the PnP Manager. The thread terminates when the NAD bus FDO is removed. The thread periodically detects existence of NICs. If a new NIC is detected, the thread creates a new NAD port PDO for the NIC and includes the newly created NAD port PDO into its own list of NAD port PDOs. The thread then invokes the PnP Manager to have the NAD port PDO recognized by the system. Removal of an MC is also detected by the thread since the thread can detect the absence of the MC of which NAD port PDO previously created would be found in the above mentioned list without the corresponding MC. If an NIC is found to have been removed from the host, the thread removes the corresponding NAD port PDO from its list and invokes the PnP Manager to remove the NAD port from the Windows 2000 system.

The software routines implemented in the NAD bus driver can be classified into five categories. The following tables list some of the routines implemented in the NAD bus driver with brief explanations.

TABLE 4

Basic functions	
DriverEntry( )	executed when the driver is initially loaded registers dispatch routines of the NAD bus driver initializes the variables used by the driver
NADBusUnload( )	recovers resources occupied by the driver when the driver is unloaded
NADBusAddDevice( )	creates NAD bus FDO initializes the value of the NAD bus FDO

TABLE 5

Dispatch functions	
NADBusCreate( )	processes the 'IRP_MJ_CREATE' IRP
NADBusClose( )	processes the 'IRP_MJ_CLOSE' IRP
NADBusPnp( )	processes the 'IRP_MJ_PNP' IRP determines whether the IRP passed is to NAD bus FDO or to NAD port PDO, and invokes NADFDOPnP( ) or NADPDOPnP( ) accordingly
NADBusPower( )	processes 'IRP_MJ_POWER' IRP

TABLE 6

NAD bus FDO related functions		
NADBusFDOPnp( )	invoked when IRP_MJ_PNP is sent to NAD bus FDO processes various minor functions according to the minor function number sent together	
	IRP_MN_START_DEVICE	transfer NAD bus FDO to 'started' state invokes NADBusStartFDO( )
	IRP_MN_QUERY_STOP_DEVICE	invoked to query if NAD bus FDO can be stopped transfer NAD bus FDO to 'stop pending' state
	IRP_MN_CANCEL_STOP_DEVICE	invoked to cancel IRP_MN_QUERY_STOP_DEVICE
	IRP_MN_STOP_DEVICE	stops NAD bus FDO transfers NAD bus FDO to 'stopped' blocks NADBusHW( ) thread
	IRP_MN_QUERY_REMOVE_DEVICE	invoked to query if NAD bus FDO can be removed from the system
	IRP_MN_CANCEL_REMOVE_DEVICE	invoked to cancel IRP_MN_QUERY_REMOVE_DEVICE
	IRP_MN_SURPRISE_REMOVAL	invoked when NAD bus FDO is removed abnormally
	IRP_MN_REMOVE_DEVICE	invoked when NAD bus FDO is removed normally
	IRP_MN_QUERY_DEVICE_RELATIONS	passes list of NAD port PDO to PnP manager
NADBusStartFdo( )	allocates resources to NAD bus FDO	
NADBusRemoveFdo( )	recovers resources occupied by NAD bus FDO removes the NADBusHW( ) thread	
NADBusGetDeviceCapabilities( )	passes DeviceCapability data dtructure to PnP manager	

TABLE 7

NAD port PDO related functions	
NADPortPDOPnp( )	processes minor functions related to PnP invoked when IRP_MJ_PNP is sent to NAD port PDO Minor functions: IRP_MN_START_DEVICE IRP_MN_QUERY_STOP_DEVICE IRP_MN_CANCEL_STOP_DEVICE IRP_MN_STOP_DEVICE IRP_MN_QUERY_REMOVE_DEVICE IRP_MN_CANCEL_REMOVE_DEVICE IRP_MN_SURPRISE_REMOVAL IRP_MN_REMOVE_DEVICE IRP_MN_QUERY_CAPABILITIES IRP_MN_QUERY_ID IRP_MN_QUERY_DEVICE_RELATIONS IRP_MN_QUERY_DEVICE_TEXT IRP_MN_QUERY_RESOURCES_REQUIREMENTS IRP_MN_QUERY_RESOURCE
NADPortPDOQueryDeviceCaps( )	returns DEVICE_CAPABILITIES data structure of NAD port
NADPortPDOQueryDeviceId( )	returns device ID, instance ID, hardware ID of NAD port
NADPortPDOQueryDeviceText( )	returns location and description of NAD port
NADPortPDOQueryDeviceRelations( )	returns target device relation value
NADPortInitializePdo( )	initialize NAD port PDO value invoked when NAD port attached to NAD bus is detected
NADPortDestroyPdo( )	removes NAD port PDO and recovers resources

TABLE 8

Function to detect NAD port	
NADBusHW( )	routine for the kernel thread to detect NAD ports attached to NAD bus periodically detects the existence of NICs if a new NIC is detected, creates NAD port PDO and invokes NADPortInitializePdo( ) if a NIC is detected to have been removed, removes NAD port PDO by invoking NADPortDestroyPdo( )

NAD Port Driver

A port driver is a lower-level driver that responds to a system-defined device control request or a driver-defined device I/O control request from a corresponding class driver.

The NAD port driver is capable of basic functions to initialize the driver and create an NAD port FDO and dispatch functions to process IRP passed down from the disk class driver layer. The IRP passed down from the disk class driver may contain a SCSI request block (SRB), which specifies the actual I/O command to be performed onto the SCSI device.

Tables 9 and 10 list the basic functions and some of the dispatch functions, of which roles are basically the same as those of the NAD bus driver described earlier, are presented with brief explanations.

TABLE 9

Basic functions	
DriverEntry( )	initializes driver registers driver functions
NADPortAddDevice( )	invoked by PnP manager to create NAD port FDO
NADPortDriverUnload( )	invoked when to remove driver recovers resources

TABLE 10

Dispatch functions for initialization, creation, and removal of the NAD port	
NADPortCreateClose( )	processes IRP__MJ__CREATE and IRP__MJ__CLOSE IRP
NADPortCleanup( )	processes IRP__MJ__CLEANUP IRP recovers resources

TABLE 10-continued

Dispatch functions for initialization, creation, and removal of the NAD port	
NADPortPnp( )	processes IRP__MJ__PNP IRP
NADPortPower( )	processes IRP__MJ__POWER IRP

In Windows 2000, a device I/O control command is included in an IRP as a device I/O control number, and the device I/O control functions are implemented in the port driver to handle the corresponding device I/O control numbers.

Besides the regular device I/O control functions in Windows 2000, additional device I/O control functions are implemented in the NAD port driver so that the NAD can be added or removed dynamically without stopping the Windows 2000 system. With conventional local disks, addition or removal of the local disk can be directly detected by the Windows 2000 system at the time of the system booting because the local disks are physically connected to the physical hardware bus. Therefore, the creation of a disk PDO for a local disk is basically initiated from the hardware interrupt at the time of the system booting. So the conventional port driver does not have to have functions that initiate addition or removal of the PDO of a disk device in the middle of the system operation.

However, in an NAD system, addition and removal of an NAD device can occur while the Windows 2000 system is running. Therefore, there should be a mechanism that can create/remove a disk PDO for the newly attached/removed disk.

The device I/O control functions implemented in the present invention handle such dynamic addition and removal of the NAD as follows. If a device control IRP that tells a new NAD hardware device is hooked up to the network is passed to the NAD port FDO, the NAD port FDO creates an NAD device PDO for the new NAD thus letting the system recognize the disk. For the removal of the NAD device, device control IRP to remove the disk is sent to and processed by NAD port FDO similarly.

The dispatch functions that handle device I/O control IRPs are summarized in the following table. Note that the device I/O control functions, NASPortFdoDeviceControl( ), NADPortPluInDevice( ) and NADPortUnpluDevice( ) are the functions particular to the present invention for the purpose of dynamic addition and removal of the NAD.

TABLE 11

NAD port device control dispatch functions	
NADPortDeviceControl( )	invoked when I/O control IRP is passed processes IRP__MJ__DEVICE__CONTROL IRP invokes NADPortFdoDeviceControl( ) for FDO control invokes NADPortPdoDeviceControl( ) for PDO control
NADPortFdoDeviceControl( )	registers new NAD device or removes an NAD device processes I/O control functions IOCTL__NADPORT__PLUGIN__HARDWARE Invokes NADPortPlugInDevice( ) to register new NAD hardware IOCTL__NADPORT__UNPLUG__HARDWARE invokes NADPortUnplugDevice( ) to remove an NAD hardware
NADPortPdoDeviceControl( )	processes I/O controls for PDO invokes I/O control functions according to the device I/O control function numbers in the IRP IOCTL__STORAGE__QUERY__PROPERTY queries NAD device property IOCTL__GET__DISK__DRIVE__GEOMETRY

TABLE 11-continued

NAD port device control dispatch functions
returns DISK_GEOMETRY data structure containing geometry information of the NAD device
IOCTL_GET_SCSI_ADDRESS
NAD device does not use SCSI address, so sets the values of PathID and TargetID 0s and returns enumeration number of NAD device to LUN in CDB

FIG. **23**A shows the flow of IRP, SRB, and CDB where the IRP is to a SCSI disk connected to a conventional hardware bus such as PCI bus in a Windows 2000 system. A disk class driver **681** passes down to a SCSI port driver **682** and a SCSI bus driver **683** an IRP that may contain an SRB. The SRB is a data structure that contains information about the requested I/O and a command descriptor block (CDB) containing a SCSI-2 standard command. Receiving the IRP with SRB from the disk class driver, the SCSI port driver **682** and the SCSI bus driver **683** deliver the CDB extracted from the SRB to the SCSI host adapter **684** to complete an actual device I/O to a SCSI disk **685**.

In a conventional local disk, disk I/O commands are delivered to a disk controller at the host adapter using the SRB data structure. But, in the NAD system of the present invention, disk input/output commands are delivered to the NIC of the host.

devices. The translation of the CDB, in such a case, can be done either at the NAD port driver or at the NAD device. If the translation is to be done at the NAD, the Windows 2000 host simply delivers a CDB to the host NIC as if it delivers a CDB to a SCSI disk. If the translation is to be done at the NAD port driver, the NAD port driver functions must translate the CDBs into a set of disk I/O commands appropriate to the specific hardware disk types.

The NAD system of the present invention supports both cases, and the type of the commands, i.e., CDB or hardware-specific commands, is determined at the time of the installation of the specific NAD. Some of the dispatch functions that process SRB with mandatory CDB operation codes are given in table 12 to show how the NAD port driver functions are implemented to handle the SRB and CDB in the present invention. Such SRB processing functions are required if the NAD port driver has to translate the CDB into a set of hardware specific I/O commands.

TABLE 12

NADPortInternalDeviceControl( )	executes SrbFunctionExecuteScsi( ) when SRB_FUNCTION_EXECUTE_SCSI is passed as the SRB function value	
SrbFunctionExecuteScsi( )	processes CDB	
	invokes CDB processing functions according to the CDB operation codes given below	
	SISCOP_TEST_UNIT_READY	tests if an NAD device is accessible
	SCSIOP_MODE_SENSE	returns configuration of NAD device
	SCSIOP_READ	reads a block from NAD
	SCSIOP_WRITE	write a block to NAD
	SCSIOP_MODE_SELECT	sets parameter to NAD
	SCSIOP_READ_CAPACITY	returns size of the next block or address of the last block
	SCSIOP_REASSIGN_BLOCKS	relocates block
	SCSCIOP_RESERVE/SCSIOP_RELEASE	changes status information
	SCSIOP_START_UNIT	starts NAD
	SCSIOP_STOP_UNIT	stops NAD
	SCSIOP_VERIFY	verifies data stored in NAD

FIG. **23**B shows the flow of IRP, SRB, and CDB (or some other types of I/O commands) in the NAD system. A disk class driver **691** passes down an IRP with an SRB to a NAD port driver **692** and a NAD bus driver **693**, which then deliver the CDB extracted from the SRB to MC **694** to complete an actual device I/O to a NAD device **695** through a network **696**.

In the present invention, the NAD system supports various types of disks including SCSI and IDE. If the NAD device is composed of SCSI disks only, the CDB is delivered as is to the host MC so that the network-attached SCSI disks can perform the requested disk I/O.

If the NAD device, however, is composed of disk devices of other type than SCSI such as IDE, the CDB must be translated into the commands that can be processed by the specific

Communication Between the Host and the NAD

Disk I/O commands in the NAD system are delivered to the host NIC instead of the local disk host adapter because the I/O should be done over the network rather than over the bus. Windows 2000 provides a Network Driver Interface Specification (NDIS), a set of specifications defined to specify network interface drivers.

FIG. **24** shows a NDIS driver layer defined in Windows 2000. It consists of a NDIS protocol driver **701** for specifying a high-level protocol to be used, a NDIS intermediate driver **702**, an NDIS miniport **703** for managing hardware specifics, and a network interface card (MC) **704**.

In the present invention, all the NAD port driver functions that deliver I/O commands to the NAD devices are implemented to deliver the commands to a NDIS (network driver interface specification) protocol driver layer through which the commands are delivered to the NAD devices over the network.

Upon receiving from the disk class driver the IRP containing an SRB or an I/O control command for specific disk I/O operation, the NAD port driver passes down a new IRP containing the corresponding CDB to the protocol driver. Then the protocol driver sends the CDB, which is the SCSI-2 standard I/O command, to the NAD device and, in turn, receives and handles the reply from the NAD device. Note here that if the host computer has to send some hardware specific I/O commands other than CDB as is pointed out in FIG. **23**B, the NAD port driver passes down an IRP containing the hardware specific commands instead of the CDB to the NDIS protocol driver.

The NDIS provides transport-independence for network vendors because all drivers that require communication over the network calls the NDIS interface to access the network, thus providing a ready solution for the communication between the host computer and the NAD devices in the present invention.

The actual protocol implemented in the protocol driver of the NDIS may adopt a standard protocol or a non-standard protocol. Since a standard protocol such as IP (Internet Protocol) involves an overhead, a non-standard protocol may be preferred in terms of performance and security. The present invention follows the NDIS specification of the Windows 2000 network system to implement a proprietary communication protocol into the NDIS protocol driver in order to provide a communication protocol between a Windows 2000 host and NAD devices to reliably handle the NAD I/O commands.

NAD Device

The technical constitution of the NAD device running under the Windows family of operating systems is the same as that of the NAD device running under the UNIX family of operating systems shown in FIG. **14**.

Advantages of the NAD System over NAS and SAN

Either running the UNIX or Windows family of operating systems, the NAD system of the present invention has numerous advantages over the NAS system and the SAN system. Unlike the NAS system that provides file storage service by way of an additional file server, the NAD device is attached to a host computer as if it is a local disk connected to the system bus of the host. Unlike the SAN system, the NAD device of the present invention is simply plugged into a network port without requiring any additional special switch or network equipment. Therefore, the NAD system provides better user convenience, system flexibility, scalability, economy, and performance.

All the disk-related operations, including formatting, partitioning, sharing, and mounting, can be done to NAD devices just as they can be done to a local disk. Since NAD devices are directly available to the host as local disks, the NAD system provides better manageability and user convenience. In the NAS system, addition, deletion, or any change to the disk configuration should be consulted to the NAS operating system through human or software intervention. In the NAD system, addition or deletion of an NAD device is instantly achieved by plugging or unplugging the NAD device to and from a network port. The NAD system even provides a superior user convenience in installing and uninstalling the disks, eliminating the need of opening and closing the case of the host computer.

The NAD system provides almost unlimited scalability to the disk capacity. The number of NAD devices that can be attached to the network is virtually unlimited, whereas the number of disks available through the NAS system is severely limited because of an economical reason and the inconvenience involved in the management of the multitudes of NAS servers.

The NAD system is intrinsically more economical than the NAS or SAN system because each NAD device does not employ file server software and other additional special hardware equipment.

Media Changeable NAD System

An NAD system of the present invention can be alternatively implemented as a media changeable storage device. A media changeable storage device is a special storage device that is physically separated two parts, one being the media containing the data and the other being the driver performing an I/O operation to the media. Floppy disk drivers, CD-ROM drivers are examples of media changeable storage devices. Whether a media is installed or not, a media changeable storage device can be registered to a host computer so that a media such as a diskette can be inserted into a driver dynamically.

Since NAD devices can be plugged in or removed from a network port dynamically, a virtual driver that uses NAD as a media can be implemented in the form of a media changeable storage device. Windows 2000 provides the changer class driver model to implement a media changeable storage device. In order to implement a media changeable NAD system, a class driver for the NAD system is implemented according to the model of the changer class driver of Windows 2000. The two lower-level drivers, i.e. the NAD port driver and the NAD bus driver, are used to implement such media changeable NAD system.

Alternative Embodiment Using Converter and Counter-Converter

Instead of using a network interface card (NIC) and new virtual host bus adapter, the network attached disk of the present invention may be implemented by providing in the host side a protocol converter that converts storage commands into data link frames containing the storage commands so that the frames can be sent through a network, and by providing in the device side a counter-converter that converts the data link frames containing the storage commands received through the network into the storage commands.

Since a converter is a specialized network interface, the converter encapsulates the I/O commands and data to data link frame so as transmit them to an I/O device through a network without the overhead of processing communication protocols in general.

Tape System, CD Juke Box

The kinds of storage devices that can be directly connected to a network using the interface of the present system are not limited to disk systems. Tape systems and CD drives use IDE or SCSI interface, the same bus interface as disk systems. For example, the present invention may be used to connect multiple CD drives directly to a network, enabling a cost-effective implementation of a CD-Juke box.

While the invention has been described with reference to preferred embodiments, it is not intended to be limited to those embodiments. It will be appreciated by those of ordinary skilled in the art that many modifications can be made to the structure and form of the described embodiments without departing from the spirit and scope of this invention.

The invention claimed is:

1. A network-attached device (NAD) access system wherein a host, having an internal host system bus and running an operating system, controls an external device through a carrying general-purpose network traffic using a certain network protocol, the system comprising:

a network interface card (NIC) installed at the host for providing a general purpose network connection between the host and the network and via the network to other devices coupled to the network;

a network-attached device (NAD) having a data storage to store data, the NAD coupled to the network for receiving device level access commands from the host in data link frames according to the certain network protocol through the network; and

a device driver, running at the host, for creating a virtual host bus adapter in software controlling the NAD through the network via the NIC, the device driver enumerating NAD that are available over the network, not directly attached to the host internal system bus, to make the host recognize the NAD as a host local device;

the virtual host bus adapter controlling the NAD in a way indistinguishable from the way as a physical host bus adapter device controls device so that the host recognizes the NAD as if it is a local device connected directly to the system bus of the host.

2. The NAD system of claim **1**, wherein the network is a wired network.

3. The NAD system of claim **1**, wherein the network is a wireless network.

4. The NAD system of claim **1**, wherein the NIC is an Ethernet card.

5. The NAD system of claim **1**, wherein the operating system is the UNIX family of operating systems.

6. The NAD system of claim **5**, wherein the device driver comprises a device file and device driver routines for the device driver to register the device driver to the host.

7. The NAD system of claim **6**, further comprising a device accessing thread for accessing the NAD device.

8. The NAD system of claim **6**, further comprising a device searching thread for searching for a device attached to the network.

9. The NAD system of claim **6**, further comprising a network connection setting thread for making a connection between the device driver and the NAD device.

10. The NAD system of claim **6**, wherein the NAD disk further comprises a plurality of individual disk partitions.

11. The NAD system of claim **10**, wherein the device driver comprises a plurality of individual device drivers, each individual device driver for controlling the individual disk partition whereby each individual disk partition is accessed as an independent local disk.

12. The NAD system of claim **10**, wherein the device driver comprises a plurality of individual partitions, each driver partition for controlling each individual disk partition.

13. The NAD system of claim **1**, further comprising a second NAD device, a second device driver, and a unified device driver combining the device driver and the second device driver so that the NAD device and the second NAD device are recognized as logically a single local device.

14. The NAD system of claim **1**, wherein the driver includes: a bus driver for creating the virtual host adapter to access the NAD device as fit is a local device connected directly to the system bus of the host; and a port driver for communicating the disk access command from the host to the NAD device through a network port.

15. The NAD system of claim **14**, wherein the operating system is the Windows family of operating systems.

16. The NAD system of claim **1**, wherein the NAD device further comprises:

a disk controller for controlling the disk; and

a network adapter for receiving the disk access command through the network.

17. The NAD system of claim **16**, wherein said one or more disks are formatted as local disks.

18. The NAD system of claim **16**, further comprising a filter program providing a utility function.

19. The NAD system of claim **18**, wherein the utility function includes the function of controlling sharing of access among multiple hosts.

20. The NAD system of claim **18**, wherein the utility function includes the function of controlling transfer of access right of one host to another.

21. The NAD system of claim **18**, wherein the utility function includes the function of providing automatic back-up of the NAD device.

22. The NAD system of claim **18**, wherein the utility function includes the function of controlling access to the NAD device.

* * * * *

EXHIBIT I

US007792923B2

(12) **United States Patent**
Kim

(10) **Patent No.:** **US 7,792,923 B2**
(45) **Date of Patent:** **Sep. 7, 2010**

(54) **DISK SYSTEM ADAPTED TO BE DIRECTLY ATTACHED TO NETWORK**

(75) Inventor: **Han-gyoo Kim**, Seoul (KR)

(73) Assignee: **Zhe Khi Pak**, Moscow (RU)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 864 days.

(21) Appl. No.: **09/974,082**

(22) Filed: **Oct. 9, 2001**

(65) **Prior Publication Data**

US 2002/0069245 A1 Jun. 6, 2002

Related U.S. Application Data

(60) Provisional application No. 60/240,344, filed on Oct. 13, 2000.

(51) **Int. Cl.**
G06F 15/16 (2006.01)

(52) **U.S. Cl.** **709/218**; 709/225; 709/226

(58) **Field of Classification Search** 709/250, 709/236, 246, 217–219, 203, 223–226; 707/204, 707/205; 710/5; 711/111
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,329,619 A 7/1994 Page et al.
5,426,427 A 6/1995 Chinnock et al.
5,455,926 A 10/1995 Keele et al.
5,463,772 A * 10/1995 Thompson et al. 707/101
5,513,314 A 4/1996 Kandasamy et al.
5,524,247 A 6/1996 Mizuno
5,566,331 A * 10/1996 Irwin et al. 707/10
5,642,337 A 6/1997 Oskay et al.
5,721,818 A 2/1998 Hanif et al.
5,774,660 A 6/1998 Brendel et al.
5,781,550 A 7/1998 Templin et al.
5,812,930 A 9/1998 Zavrel
5,838,916 A 11/1998 Domenikos et al.
5,845,104 A 12/1998 Rao
5,889,942 A 3/1999 Orenshteyn
5,987,523 A 11/1999 Hind et al.

(Continued)

FOREIGN PATENT DOCUMENTS

WO WO99/03297 7/1999

(Continued)

OTHER PUBLICATIONS

Japanese Office Action, App. No. 555298/2002, Jan. 9, 2007.

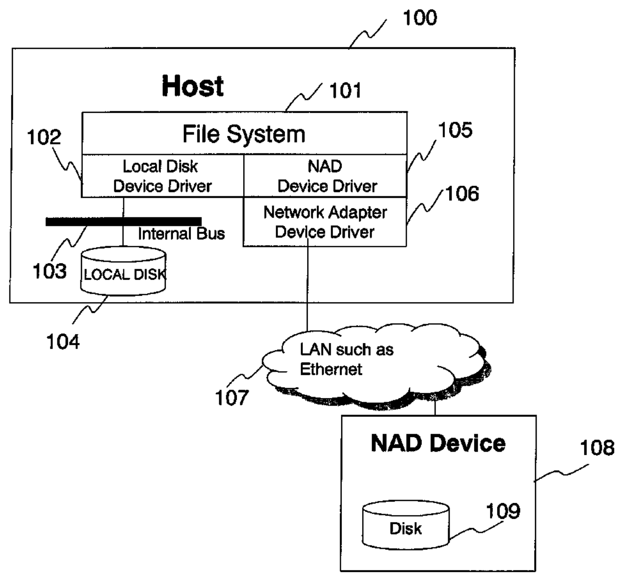
(Continued)

Primary Examiner—Faruk Hamza
(74) *Attorney, Agent, or Firm*—Rothwell, Figg, Ernst & Manbeck, P.C.

(57) **ABSTRACT**

A network-attached disk (NAD) system is disclosed that includes an NAD device for receiving a disk access command from a host through a network, and a device driver at the host for controlling the NAD device through the network, where the device driver creates a virtual host bus adapter so that the host recognizes the NAD device as if it is a local device to the host. The host may run the UNIX or Windows family of operating systems. The NAD device includes a disk for storing data, a disk controller for controlling the disk, and a network adapter for receiving a disk access command from the host through a network port.

22 Claims, 24 Drawing Sheets



U.S. PATENT DOCUMENTS

5,987,627	A	*	11/1999	Rawlings, III	714/48
5,999,808	A		12/1999	Ladue	
6,047,307	A		4/2000	Radko	
6,085,234	A		7/2000	Pitts et al.	
6,128,644	A		10/2000	Nozaki	
6,128,690	A		10/2000	Purcell et al.	
6,167,490	A		12/2000	Levy et al.	
6,175,869	B1		1/2001	Ahuja et al.	
6,216,202	B1	*	4/2001	D'Errico	711/112
6,314,465	B1		11/2001	Paul et al.	
6,317,775	B1		11/2001	Colie et al.	
6,327,594	B1		12/2001	Van Huben et al.	
6,345,300	B1		2/2002	Bakashi et al.	
6,347,095	B1		2/2002	Tang et al.	
6,356,915	B1	*	3/2002	Chtchetkine et al.	707/200
6,360,265	B1		3/2002	Falck et al.	
6,366,988	B1	*	4/2002	Skiba et al.	711/165
6,389,432	B1	*	5/2002	Pothapragada et al.	707/205
6,393,569	B1		5/2002	Orenshteyn	
6,404,766	B1		6/2002	Kitai et al.	
6,421,753	B1	*	7/2002	Hoese et al.	710/305
6,449,647	B1		9/2002	Colby et al.	
6,470,389	B1		10/2002	Chung et al.	
6,510,164	B1		1/2003	Ramaswamy et al.	
6,518,965	B2		2/2003	Dye et al.	
6,523,066	B1		2/2003	Montroy et al.	
6,529,996	B1	*	3/2003	Nguyen et al.	711/114
6,539,446	B1		3/2003	Chan	
6,578,111	B1		6/2003	Damron et al.	
6,594,677	B2	*	7/2003	Davis et al.	707/204
6,598,068	B1		7/2003	Clark	
6,609,167	B1		8/2003	Bastiani et al.	
6,647,016	B1		11/2003	Isoda et al.	
6,732,104	B1	*	5/2004	Weber	707/10
6,760,783	B1		7/2004	Berry	
6,807,581	B1	*	10/2004	Starr et al.	709/250
6,823,458	B1		11/2004	Lee et al.	
6,834,326	B1	*	12/2004	Wang et al.	711/114
6,894,981	B1		5/2005	Coile et al.	
6,941,576	B2		9/2005	Amit	
7,010,303	B2		3/2006	Lewis et al.	
7,069,312	B2		6/2006	Kostic et al.	
7,069,350	B2		6/2006	Fujita et al.	
7,076,690	B1		7/2006	Todd et al.	
7,124,128	B2		10/2006	Springer et al.	
7,251,704	B2		7/2007	Solomon et al.	
7,254,578	B2		8/2007	Devarakonda et al.	
7,277,955	B2		10/2007	Elliott	
7,376,133	B2		5/2008	Gettala et al.	
7,383,229	B2		6/2008	Jacoby	
2003/0014569	A1		1/2003	Kim	
2003/0018403	A1		1/2003	Braun et al.	
2003/0028614	A1		2/2003	Jeon	
2003/0172149	A1		9/2003	Edsall et al.	
2003/0225834	A1		12/2003	Lee et al.	
2004/0068563	A1		4/2004	Ahuja et al.	
2004/0117813	A1		6/2004	Karaoguz et al.	
2004/0220933	A1		11/2004	Walker	
2005/0042591	A1		2/2005	Bloom et al.	
2005/0110768	A1		5/2005	Marriott et al.	
2005/0149682	A1		7/2005	Kim	
2005/0193017	A1		9/2005	Kim	
2005/0193189	A1		9/2005	Kim	
2006/0004935	A1		1/2006	Seto et al.	
2006/0010287	A1		1/2006	Kim	
2006/0045130	A1		3/2006	Kim	
2006/0067356	A1		3/2006	Kim	
2006/0069884	A1		3/2006	Kim	
2006/0155805	A1		7/2006	Kim	
2007/0008988	A1		1/2007	Kim	

FOREIGN PATENT DOCUMENTS

WO	WO00/29529	5/2000
WO	WO 00/29529 A2	5/2000

OTHER PUBLICATIONS

Japanese Office Action, App. No. 513300/2003, Feb. 6, 2007.
Blunden et al., "Storage Network Virtualization: What's it all about?", ibm.com/redbooks, XP-002286341, pp. 1-110, Dec. 2000.
Klein, Yaron, "Storage Virtualization with iSCSI Protocol", Internet Draft, XP-015030964, pp. 1-15, Nov. 2, 2000.
Schulz, Greg, "SAN and NAS; Complementary Technologies", http://www.mti.com/white_papers/WP20002.pdf, XP-002201566, pp. 1-7, May 1, 2000.
Supplementary European Search Report, Application No. 01272932. 3, 4 pages, Nov. 5, 2007.
EP Communication 94(3) issued in co-pending European Application No. 01 272 932.3—2413 (Issued Aug. 3, 2009) (9 pages).
Yaron Klein, Sanrad, "Storage Virtualization with iSCSI Protocol; draft-klein-ips-virt-00.txt" (Nov. 2, 2000) (15 pages).
Blunden, Mark, et al, "Storage Networking Virtualization What's it all about?" IBM Redbooks (Dec. 2000) (124 pages).
Schulz,Greg, "SAN and NAS; Complementary Technologies—SAN and NAS provide Storage and Data Sharing" Internet Citation (May 1, 2000) (11 pages).

* cited by examiner

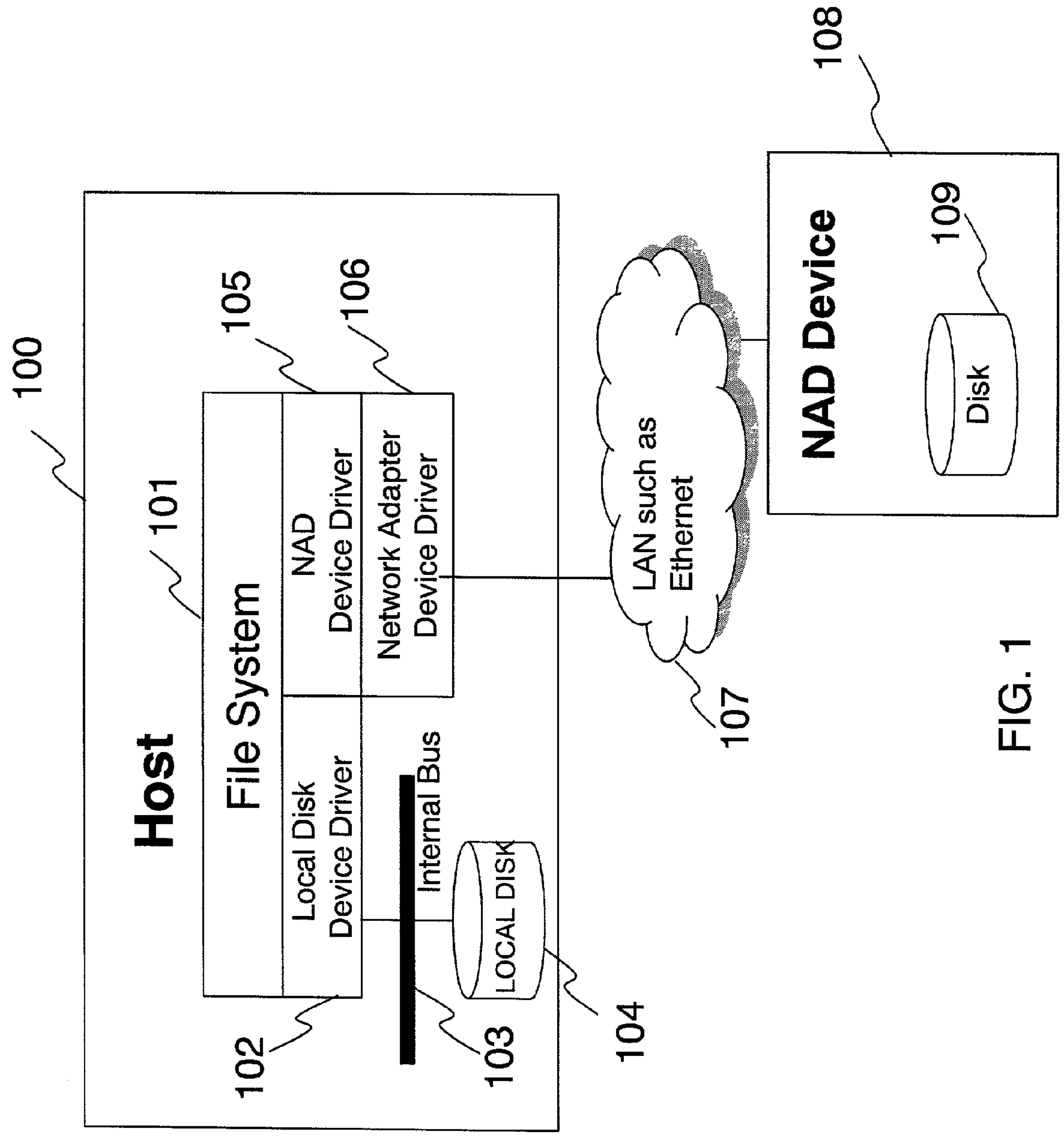

Host
File System
Local Disk
Device Driver
NAD
Device Driver
Network Adapter
Device Driver
Internal Bus
LOCAL DISK
LAN such as
Ethernet
NAD Device
Disk
100
101
102
103
104
105
106
107
108
109

FIG. 1

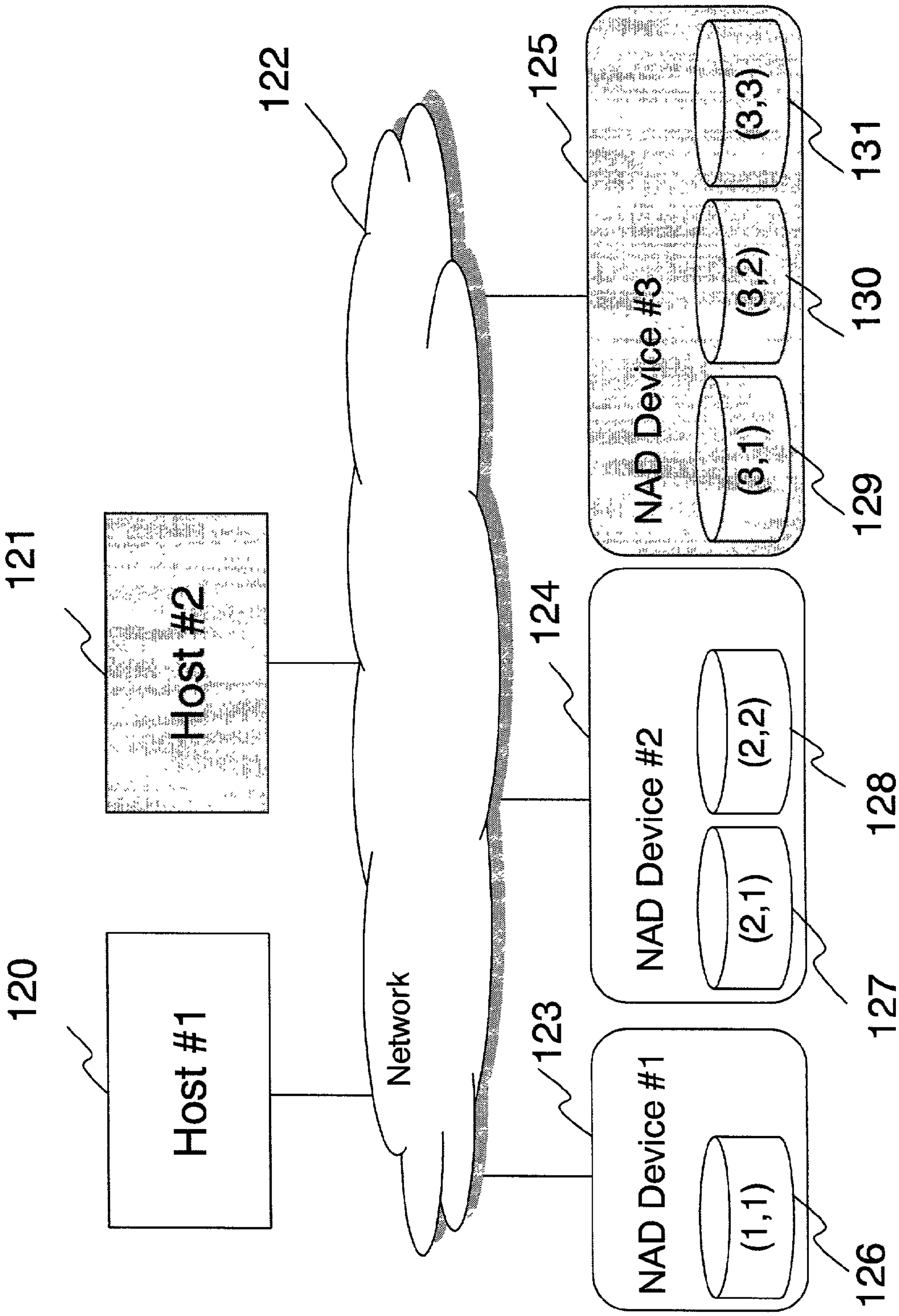

120
Host #1
121
Host #2
122
Network
123
NAD Device #1
(1,1)
126
124
NAD Device #2
(2,1)
(2,2)
127
128
125
NAD Device #3
(3,1)
(3,2)
(3,3)
129
130
131

FIG. 2

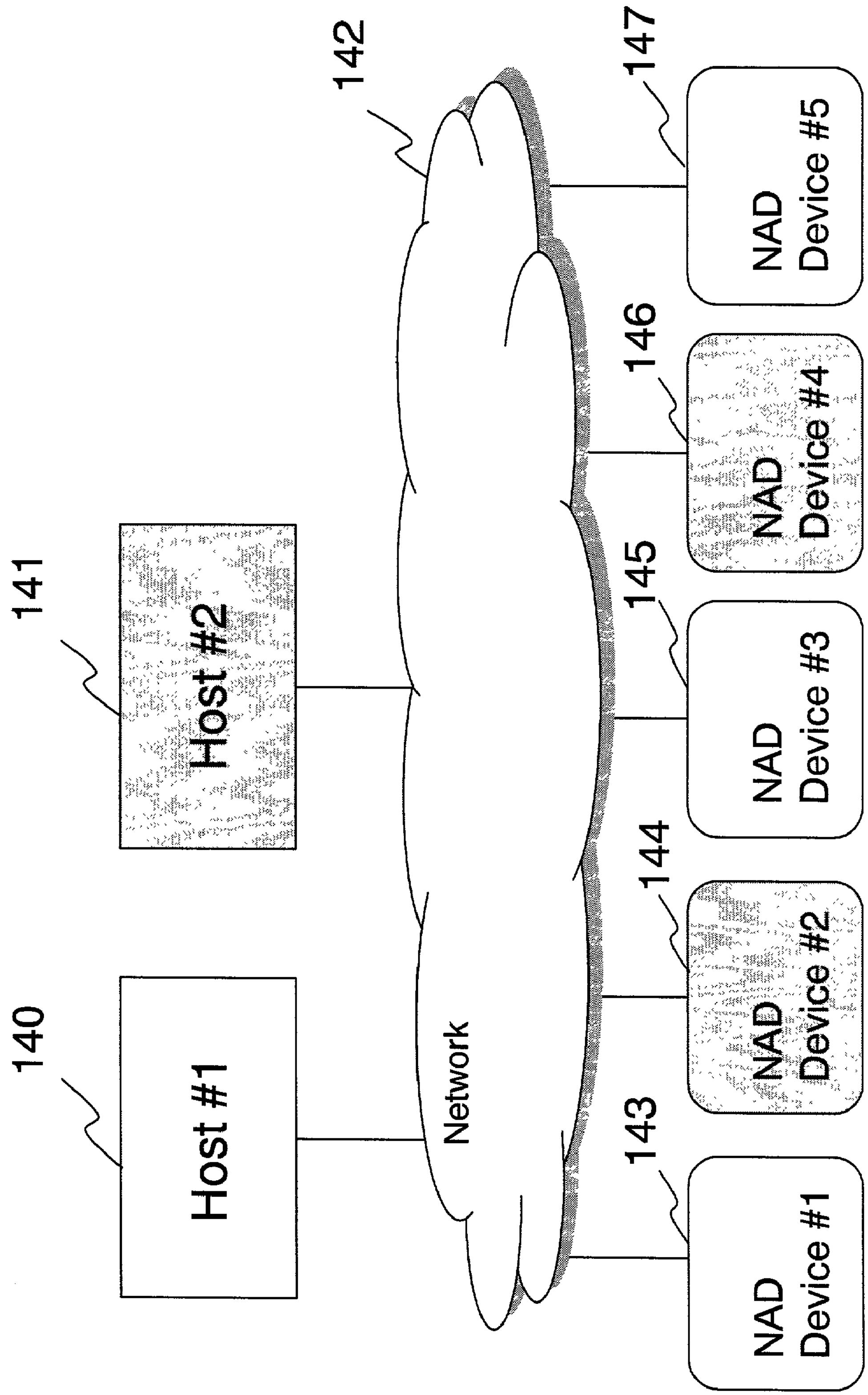
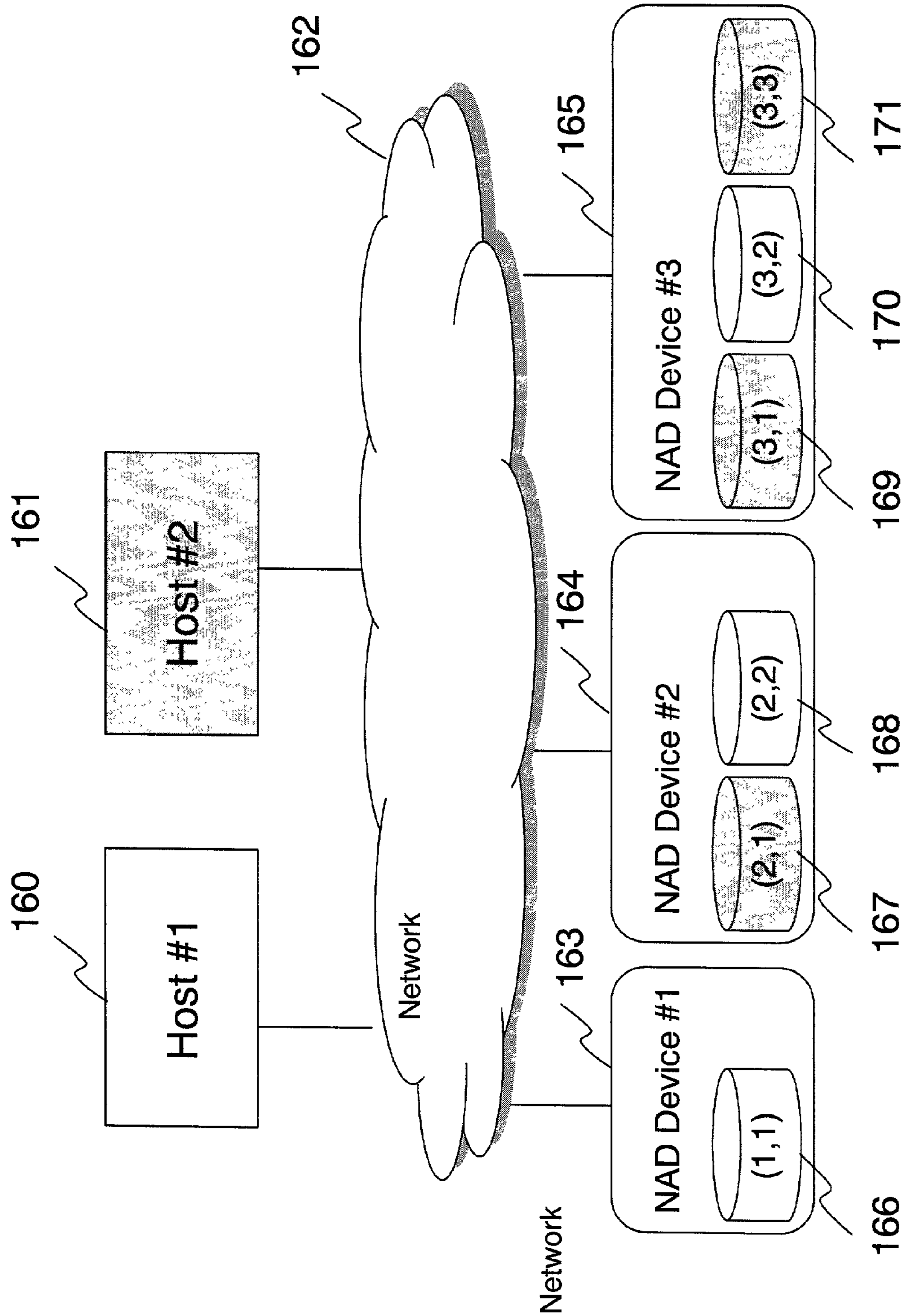

140
Host #1
141
Host #2
142
Network
143
NAD Device #1
144
NAD Device #2
145
NAD Device #3
146
NAD Device #4
147
NAD Device #5

FIG. 3



160
Host #1
161
Host #2
162
Network
163
164
165
NAD Device #1
NAD Device #2
NAD Device #3
(1,1)
(2,1)
(2,2)
(3,1)
(3,2)
(3,3)
166
167
168
169
170
171
Network

FIG. 4

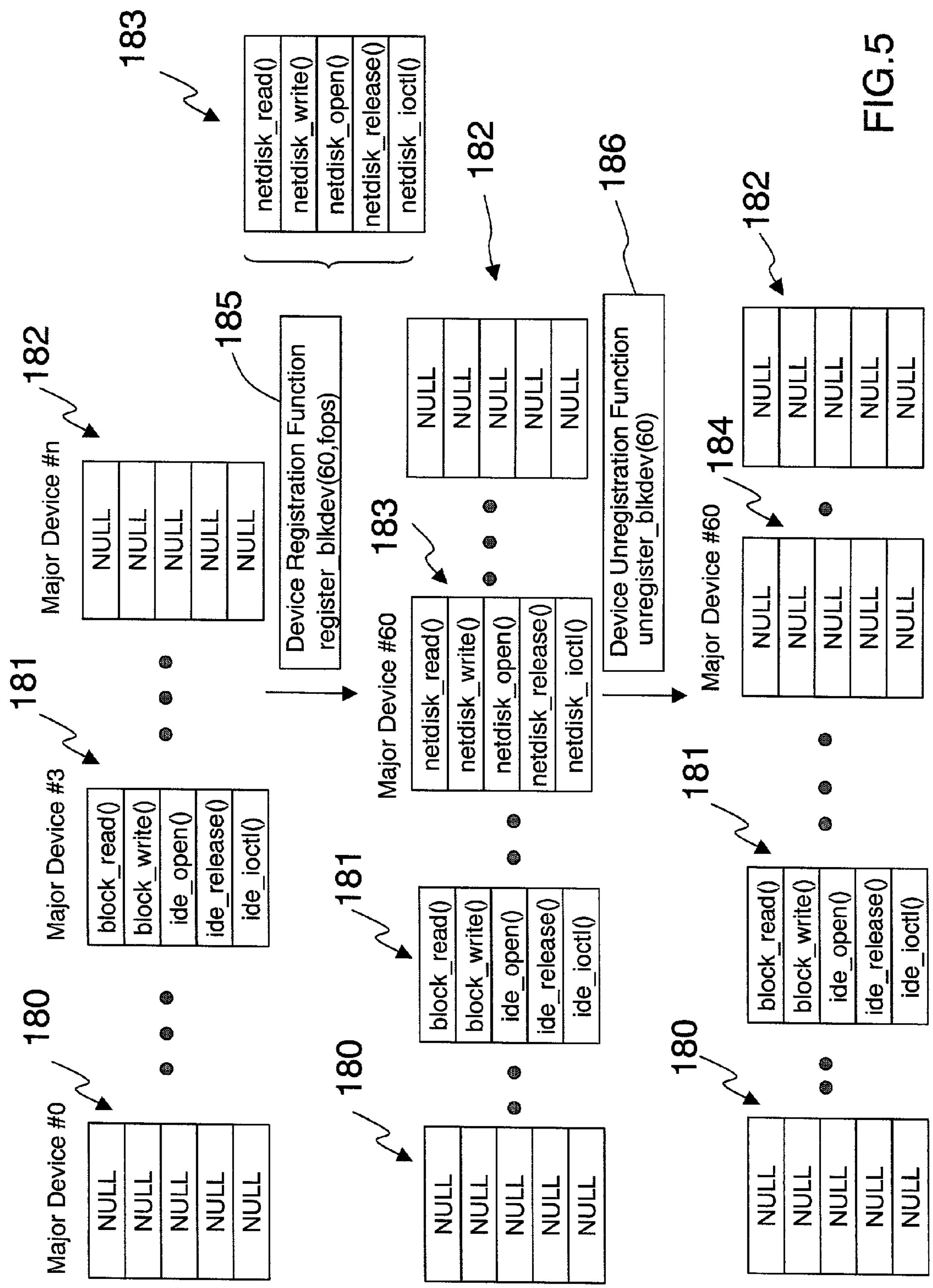

Major Device #0
NULL
NULL
NULL
NULL
NULL
180
Major Device #3
block_read()
block_write()
ide_open()
ide_release()
ide_ioctl()
181
Major Device #n
NULL
NULL
NULL
NULL
NULL
182
183
netdisk_read()
netdisk_write()
netdisk_open()
netdisk_release()
netdisk_ioctl()
185
Device Registration Function
register_blkdev(60,fops)
180
181
Major Device #60
netdisk_read()
netdisk_write()
netdisk_open()
netdisk_release()
netdisk_ioctl()
183
182
186
Device Unregistration Function
unregister_blkdev(60)
180
181
Major Device #60
184
182

FIG.5

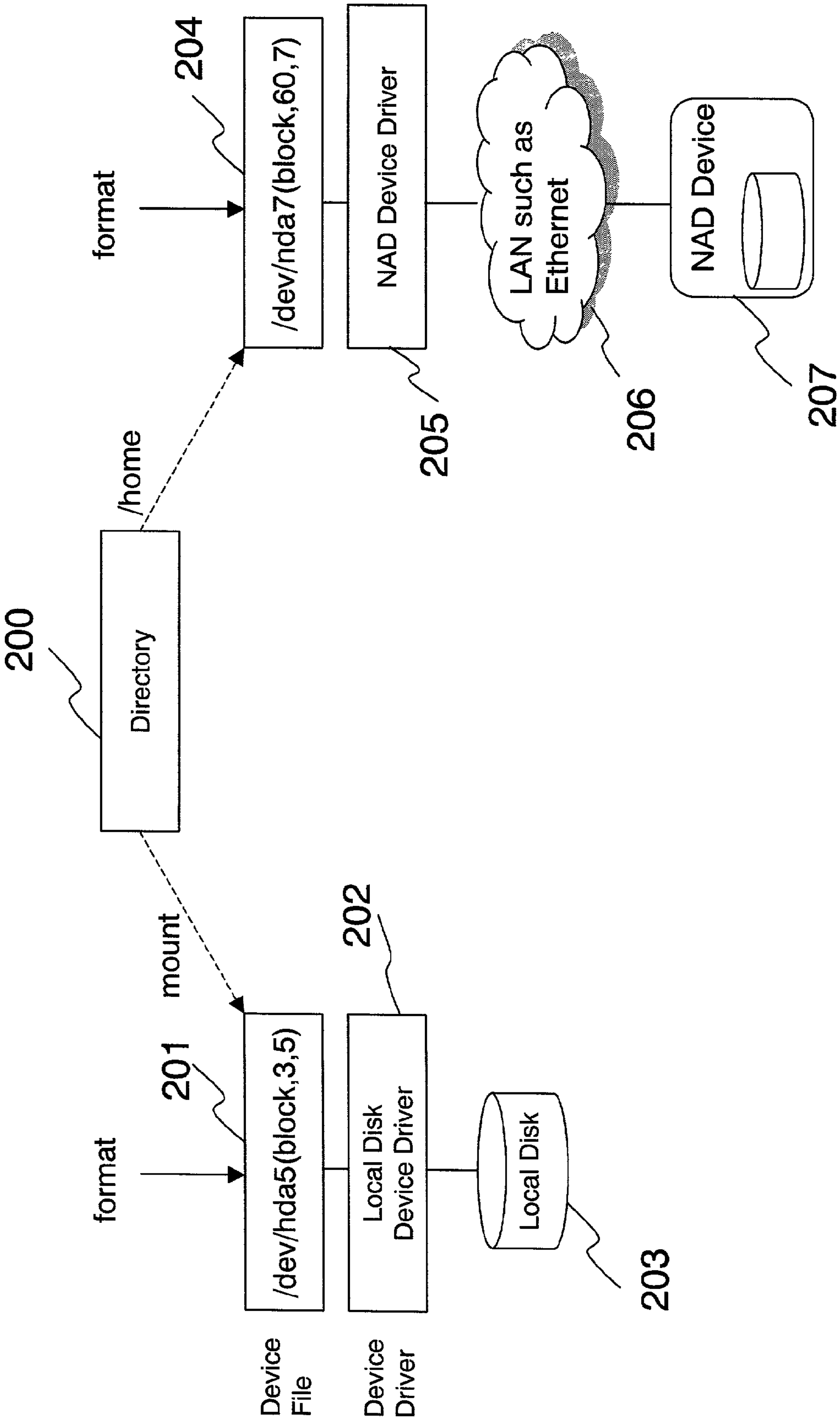
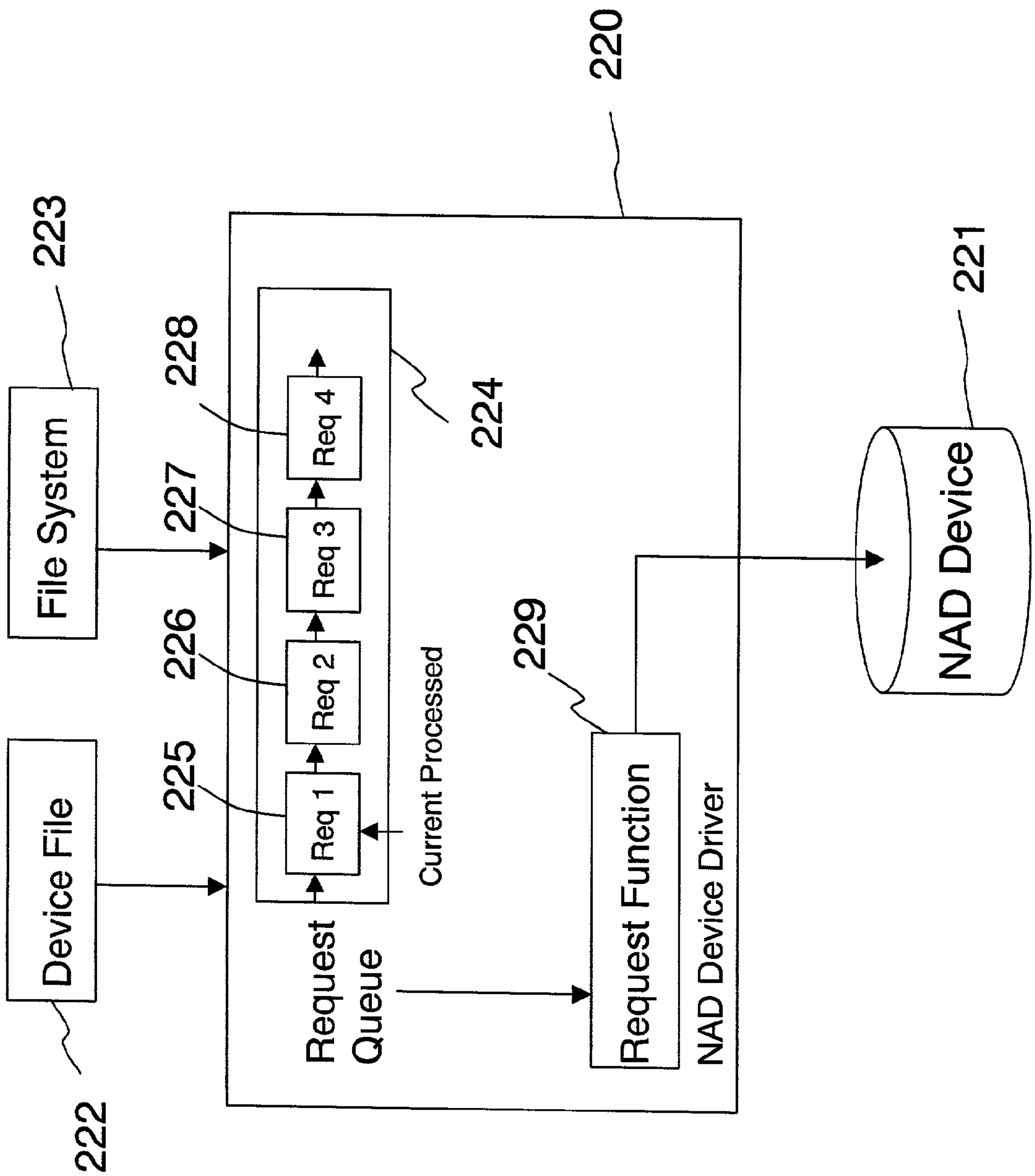

format
204
/dev/nda7(block,60,7)
NAD Device Driver
205
LAN such as Ethernet
206
NAD Device
207
/home
200
Directory
mount
format
201
/dev/hda5(block,3,5)
202
Local Disk Device Driver
Local Disk
203
Device File
Device Driver

FIG. 6

FIG. 7


220
223
File System
Device File
222
228
227
226
225
Req 4
Req 3
Req 2
Req 1
224
Current Processed
Request Queue
229
Request Function
NAD Device Driver
221
NAD Device

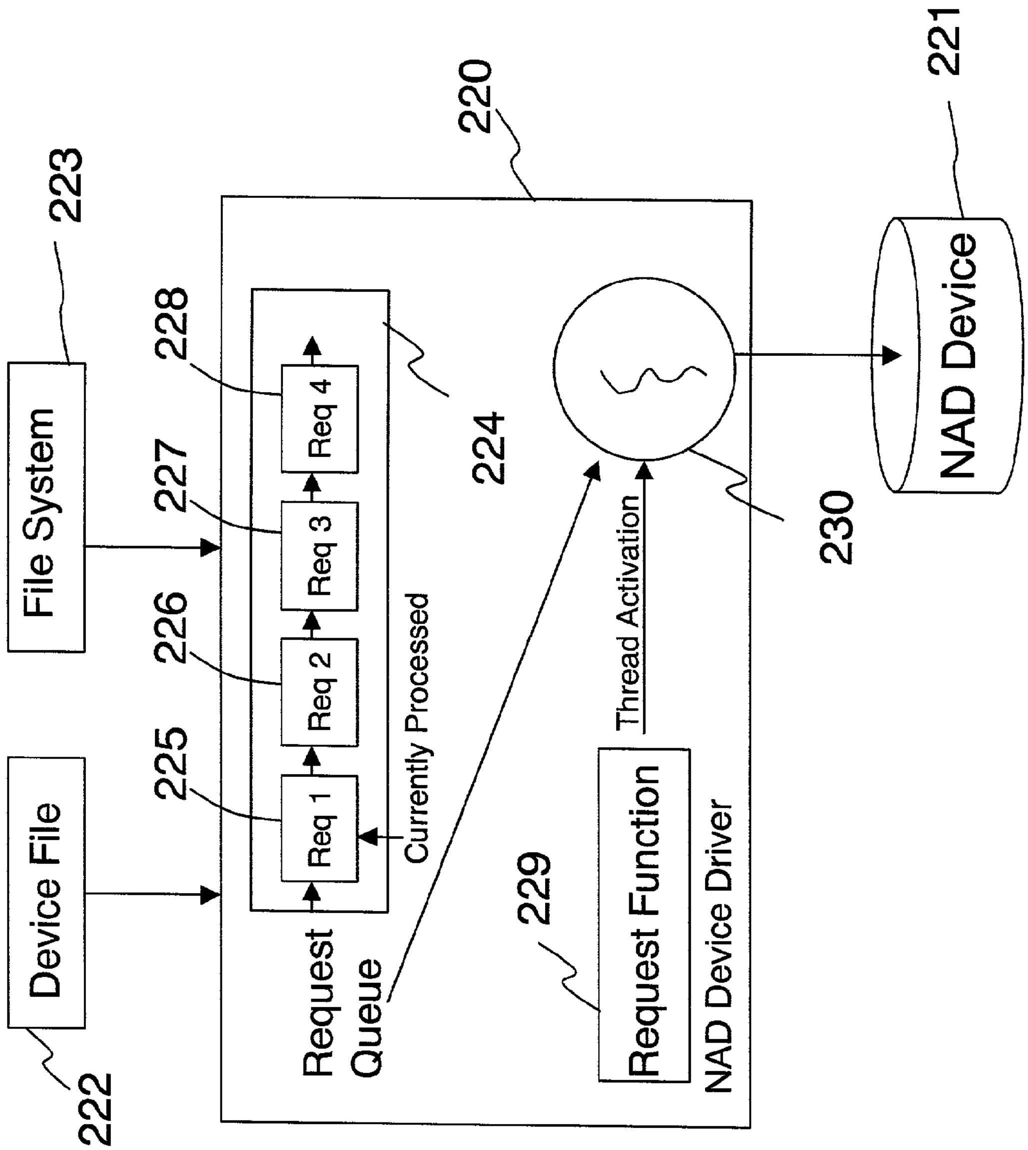

File System
Device File
223
222
220
221
Request Queue
Req 1
Req 2
Req 3
Req 4
225
226
227
228
224
Currently Processed
229
Request Function
Thread Activation
NAD Device Driver
230
NAD Device

FIG. 8

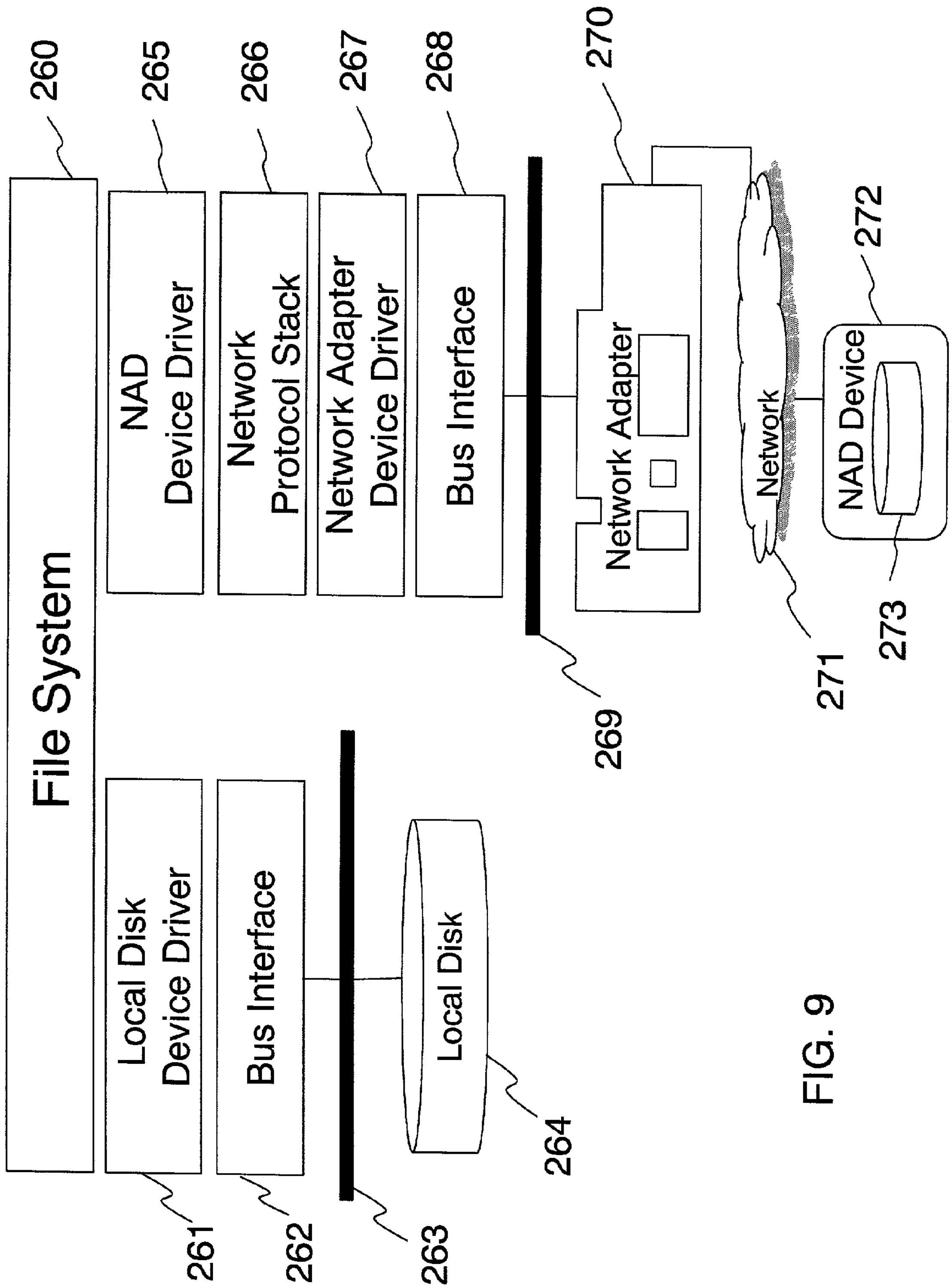

File System
NAD Device Driver
Network Protocol Stack
Network Adapter Device Driver
Bus Interface
Network Adapter
Network
NAD Device
Local Disk Device Driver
Bus Interface
Local Disk
260
265
266
267
268
270
269
271
272
273
261
262
263
264

FIG. 9

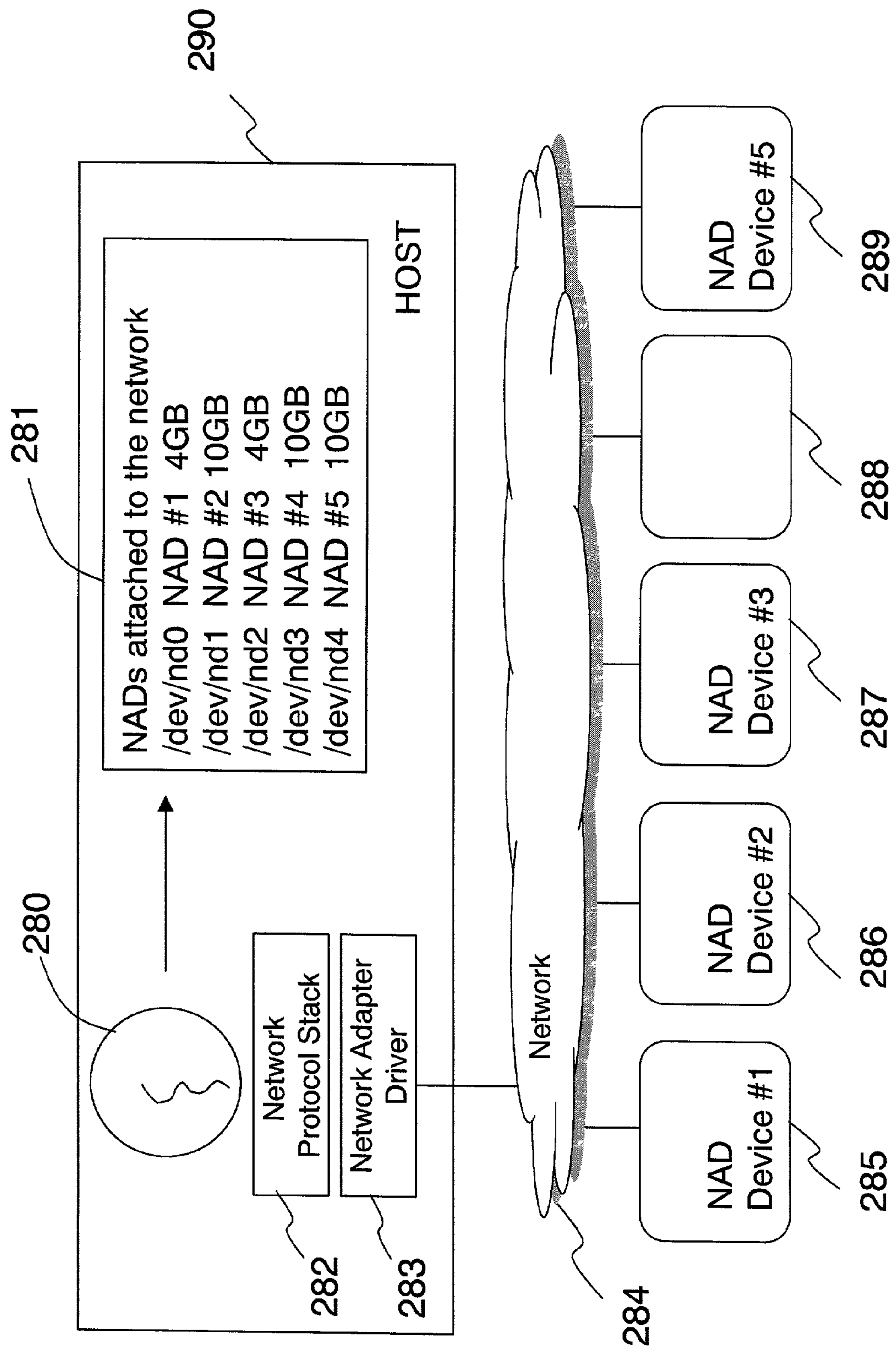

290
281
280
NADs attached to the network
/dev/nd0 NAD #1 4GB
/dev/nd1 NAD #2 10GB
/dev/nd2 NAD #3 4GB
/dev/nd3 NAD #4 10GB
/dev/nd4 NAD #5 10GB
HOST
Network Protocol Stack
Network Adapter Driver
282
283
Network
284
NAD Device #1
NAD Device #2
NAD Device #3
NAD Device #5
285
286
287
288
289

FIG. 10

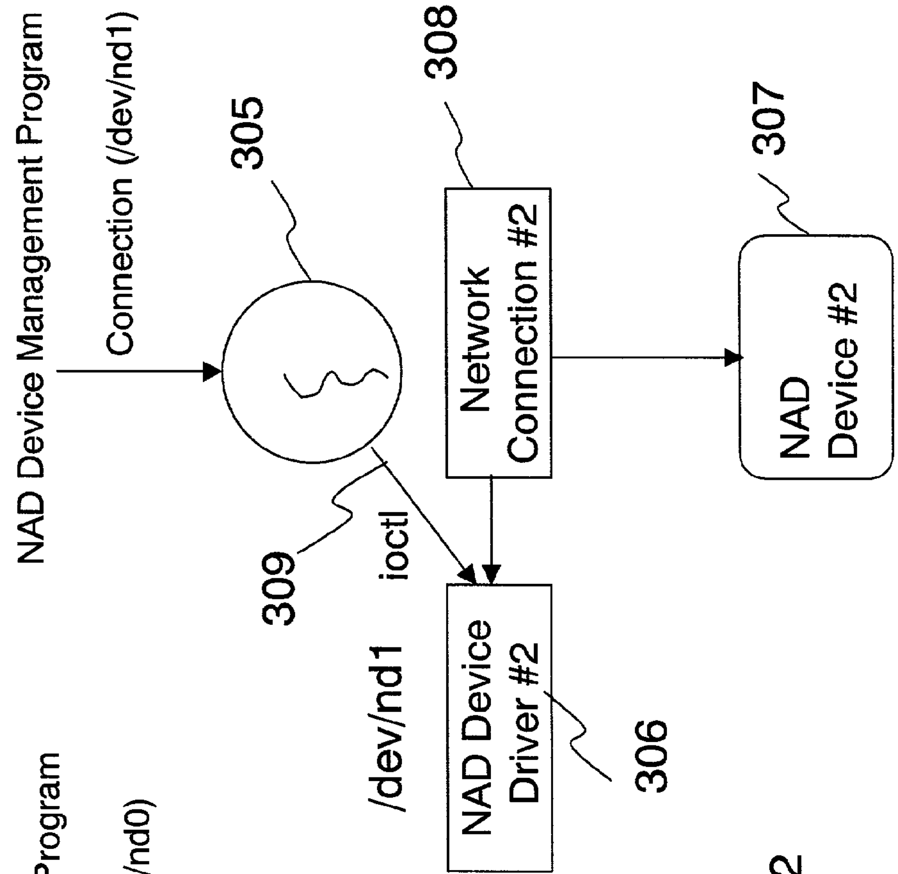


FIG. 11B

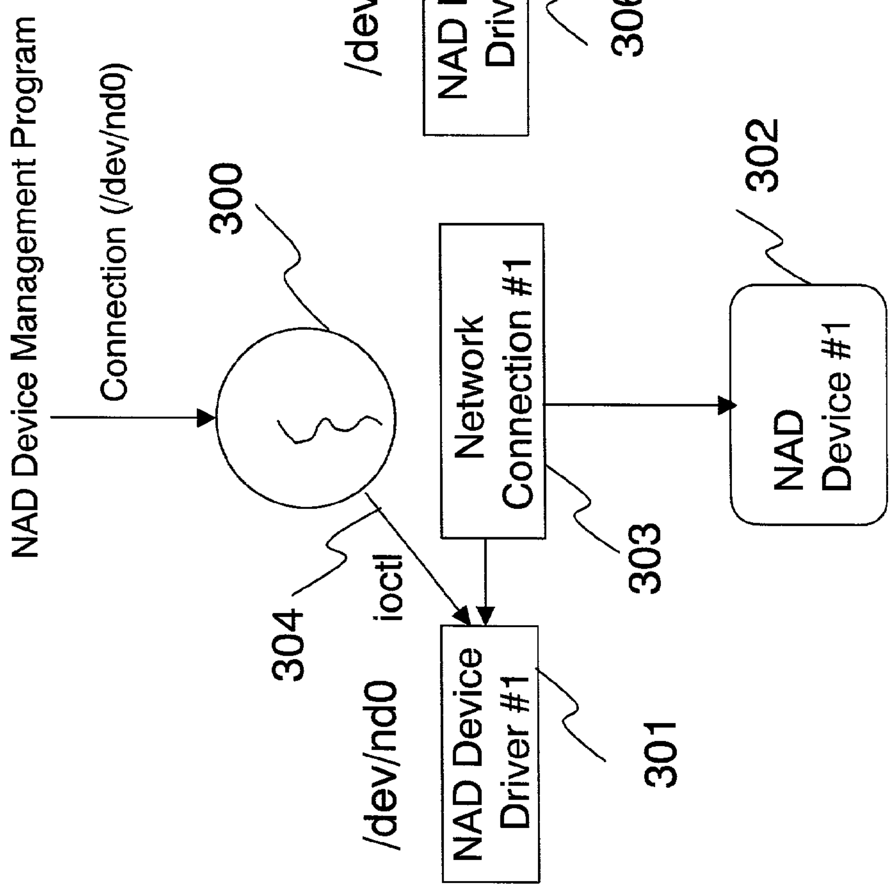


FIG. 11A

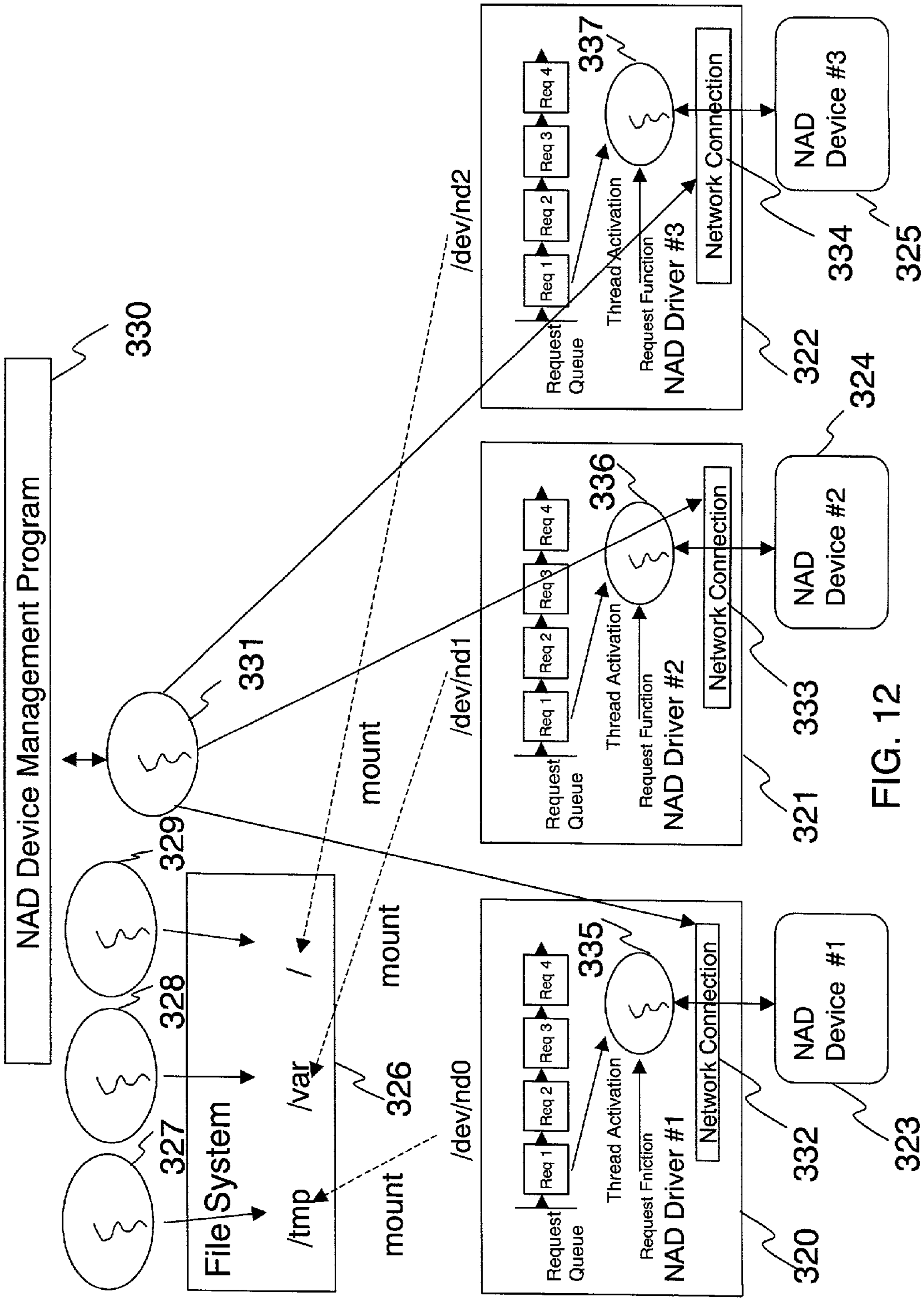

NAD Device Management Program
330
331
327
328
329
File System
/tmp
/var
/
326
mount
mount
mount
/dev/nd0
/dev/nd1
/dev/nd2
Request Queue
Req 1
Req 2
Req 3
Req 4
Thread Activation
Request Function
NAD Driver #1
NAD Driver #2
NAD Driver #3
Network Connection
335
336
337
320
321
322
332
333
334
NAD Device #1
NAD Device #2
NAD Device #3
323
324
325

FIG. 12

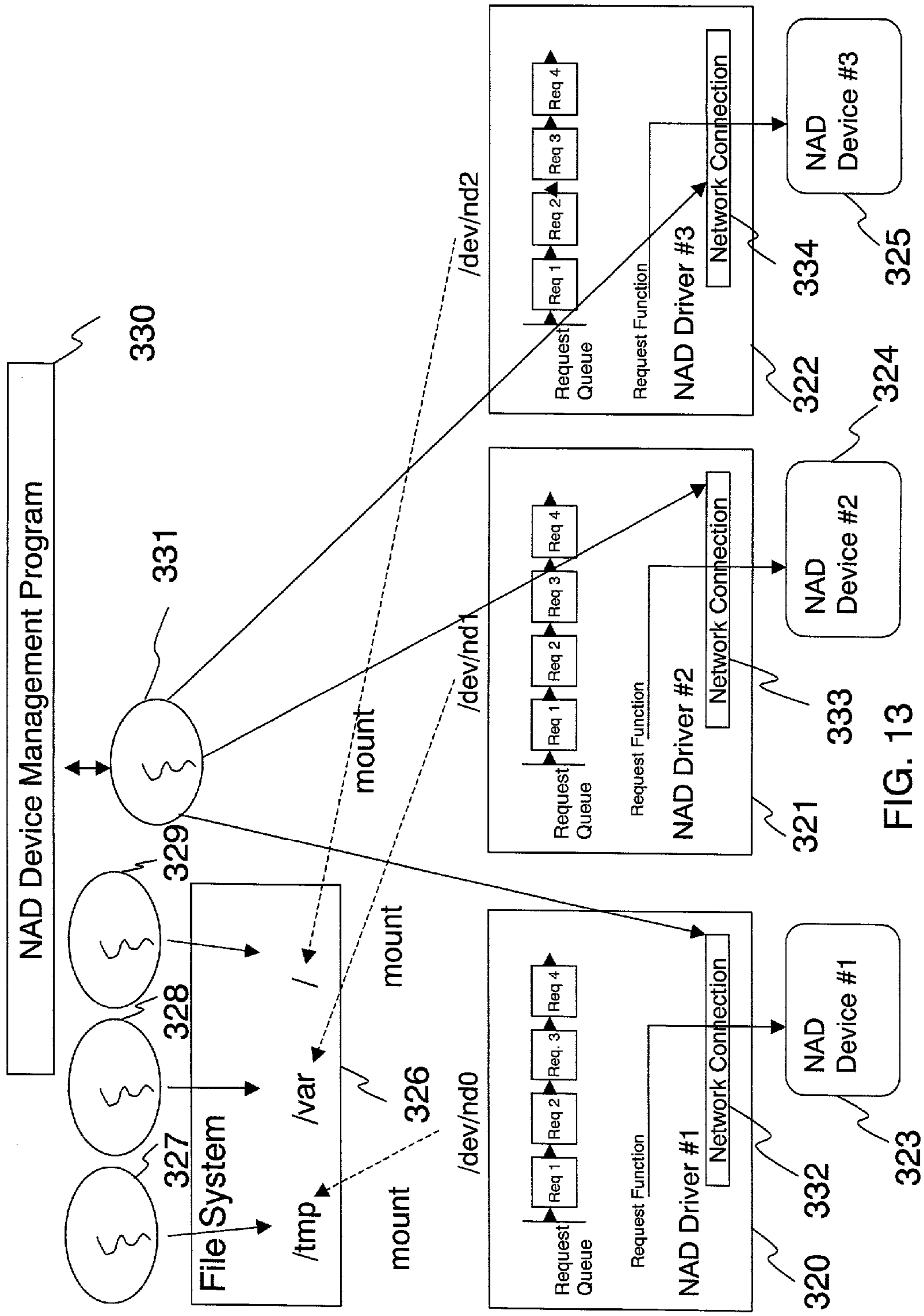

NAD Device Management Program
330
331
329
328
327
File System
/tmp
/var
/
326
mount
mount
mount
/dev/nd0
/dev/nd1
/dev/nd2
Request Queue
Req 1
Req 2
Req. 3
Req 3
Req 4
Request Function
NAD Driver #1
NAD Driver #2
NAD Driver #3
Network Connection
320
321
322
332
333
334
NAD Device #1
NAD Device #2
NAD Device #3
323
324
325

FIG. 13

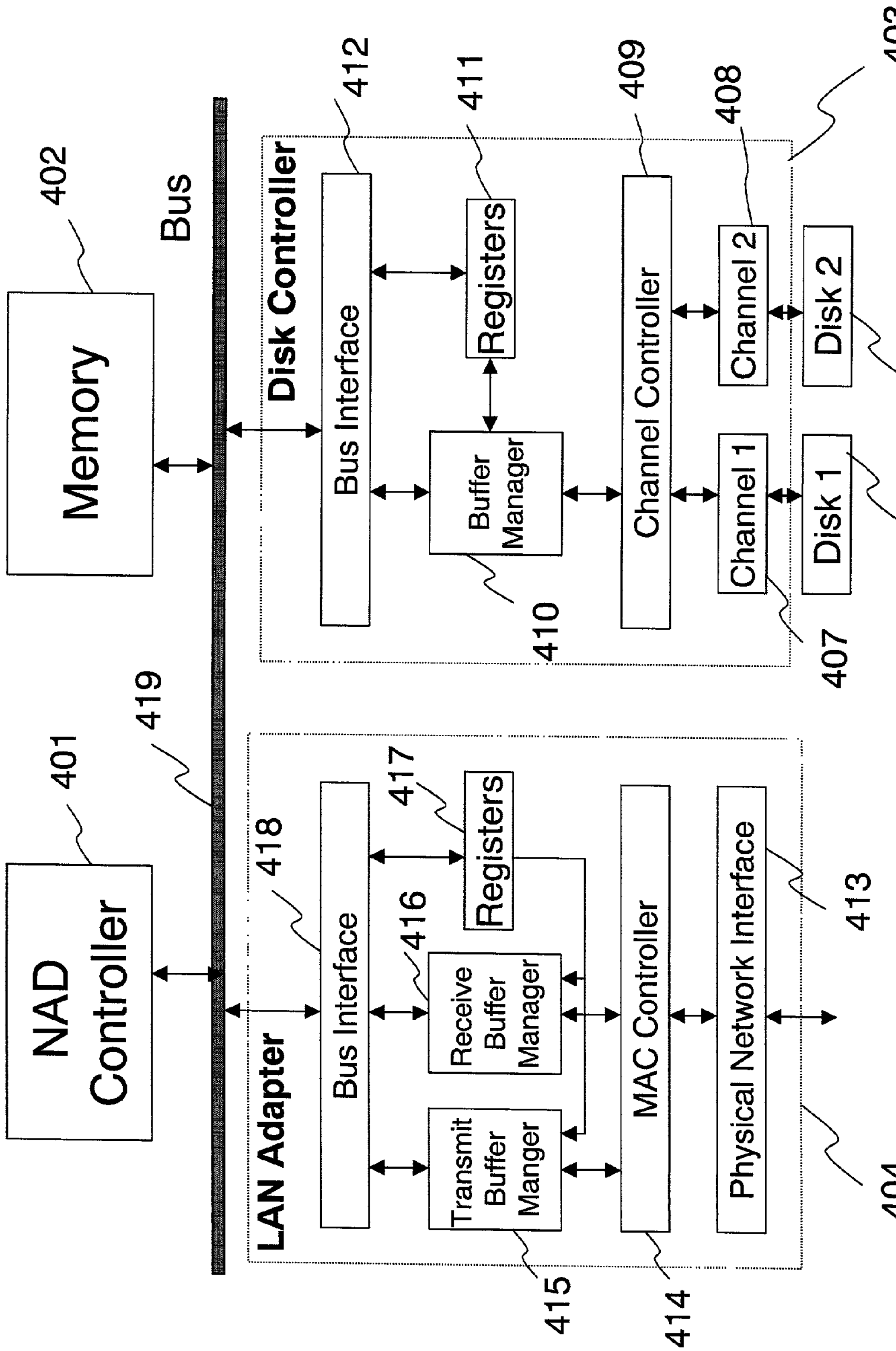

402
Memory
Bus
Disk Controller
412
Bus Interface
411
Registers
Buffer Manager
410
409
Channel Controller
408
Channel 2
Channel 1
407
Disk 2
Disk 1
406
405
403
401
NAD Controller
419
LAN Adapter
418
Bus Interface
417
Registers
416
Receive Buffer Manager
Transmit Buffer Manger
415
MAC Controller
414
Physical Network Interface
413
404

FIG. 14

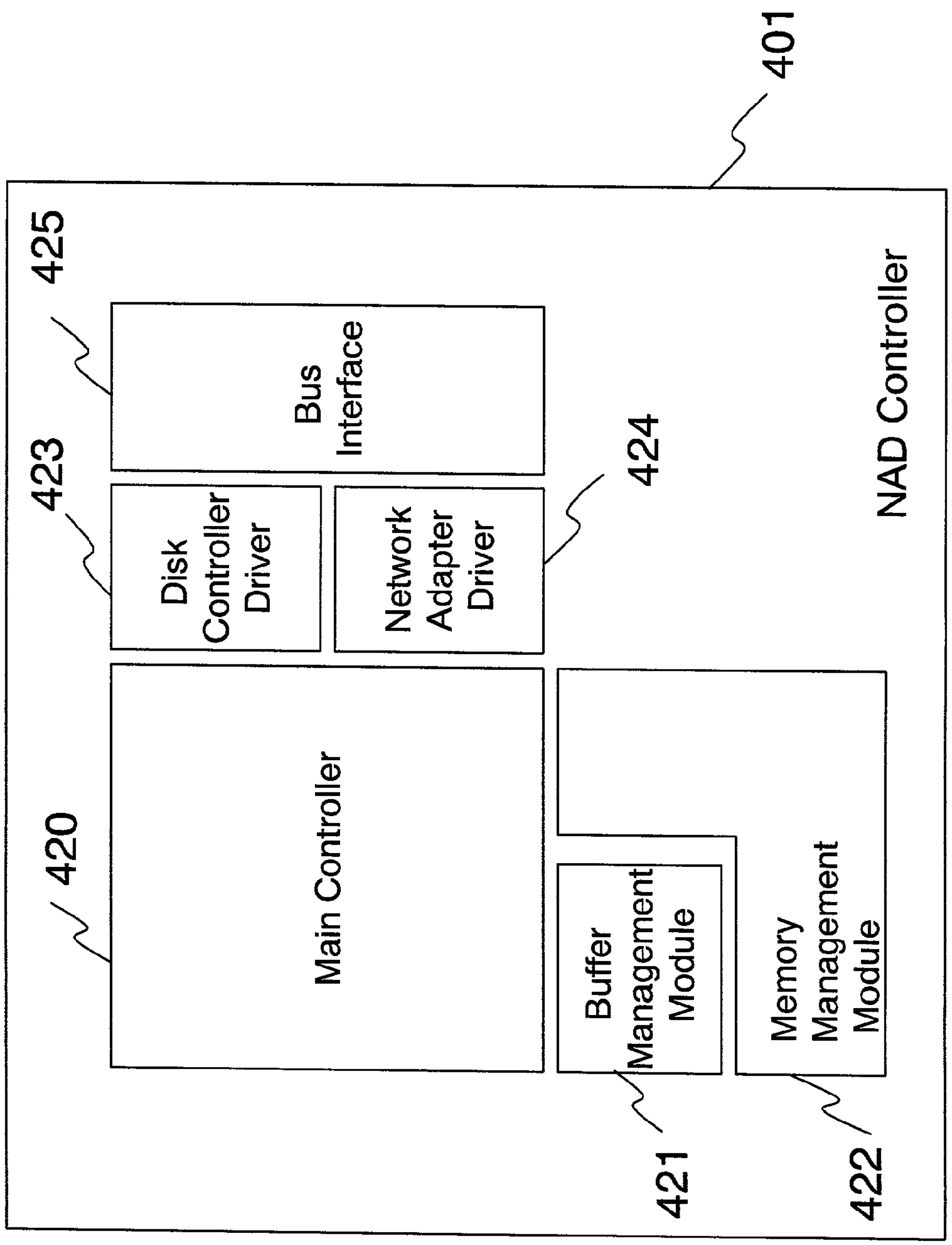

401
425
Bus Interface
423
Disk Controller Driver
424
Network Adapter Driver
NAD Controller
420
Main Controller
Buffer Management Module
Memory Management Module
421
422

FIG. 15

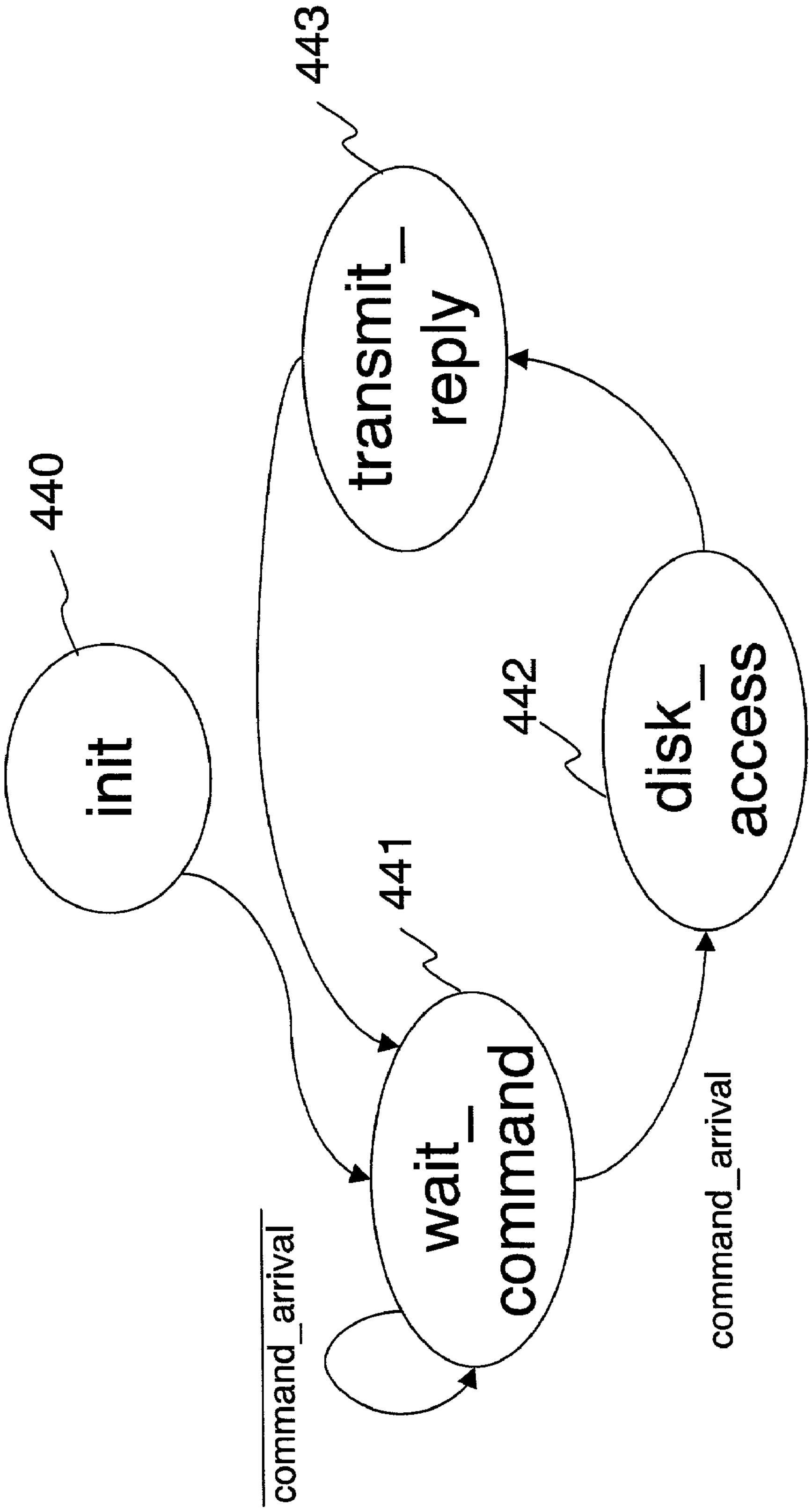

440
init
441
wait_
command
command_arrival
command_arrival
442
disk_
access
443
transmit_
reply

FIG. 16

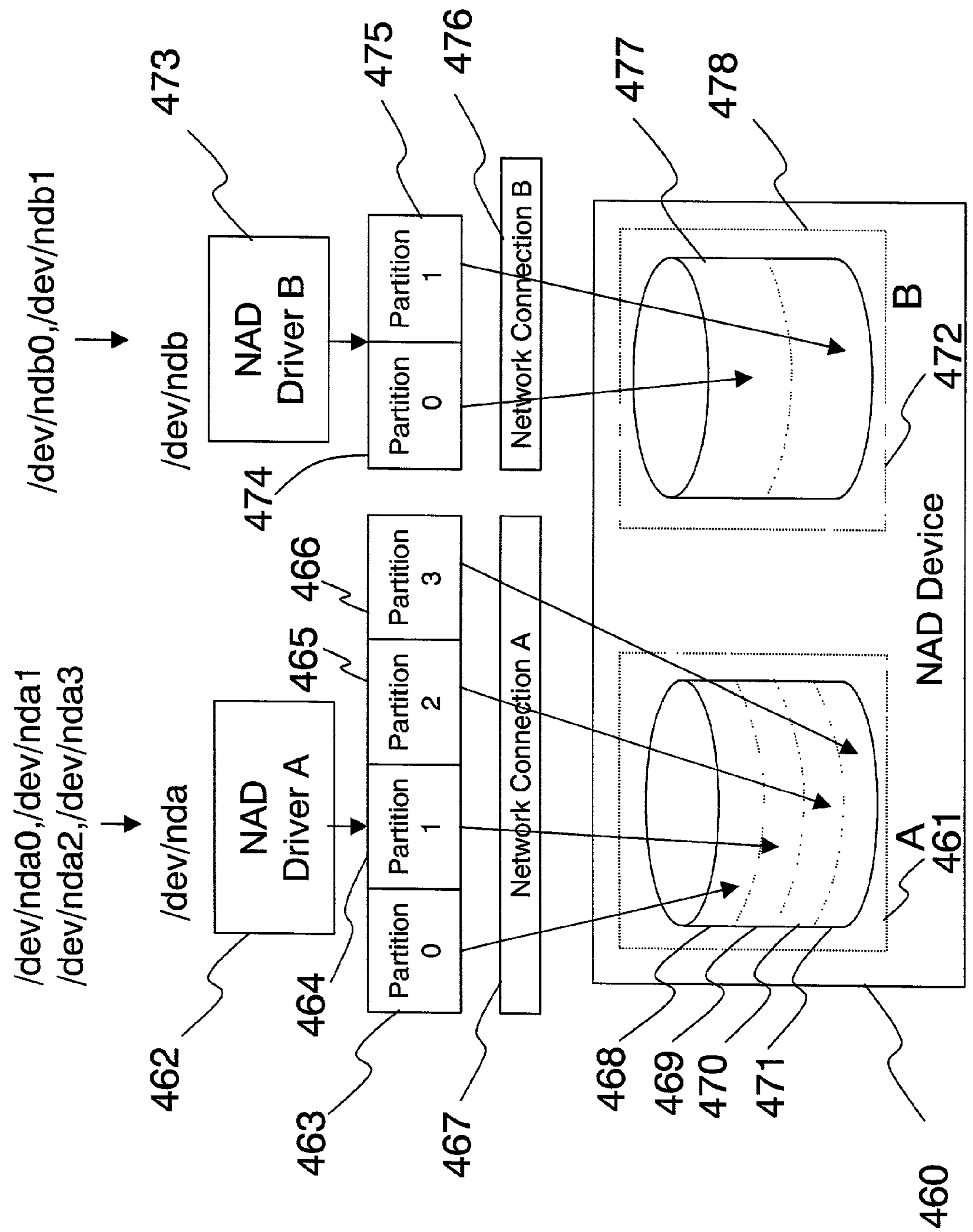

/dev/nda0,/dev/nda1
/dev/nda2,/dev/nda3
/dev/nda
NAD
Driver A
/dev/ndb0,/dev/ndb1
/dev/ndb
NAD
Driver B
Partition 0
Partition 1
Partition 2
Partition 3
Partition 0
Partition 1
Network Connection A
Network Connection B
A
B
NAD Device
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478

FIG. 17

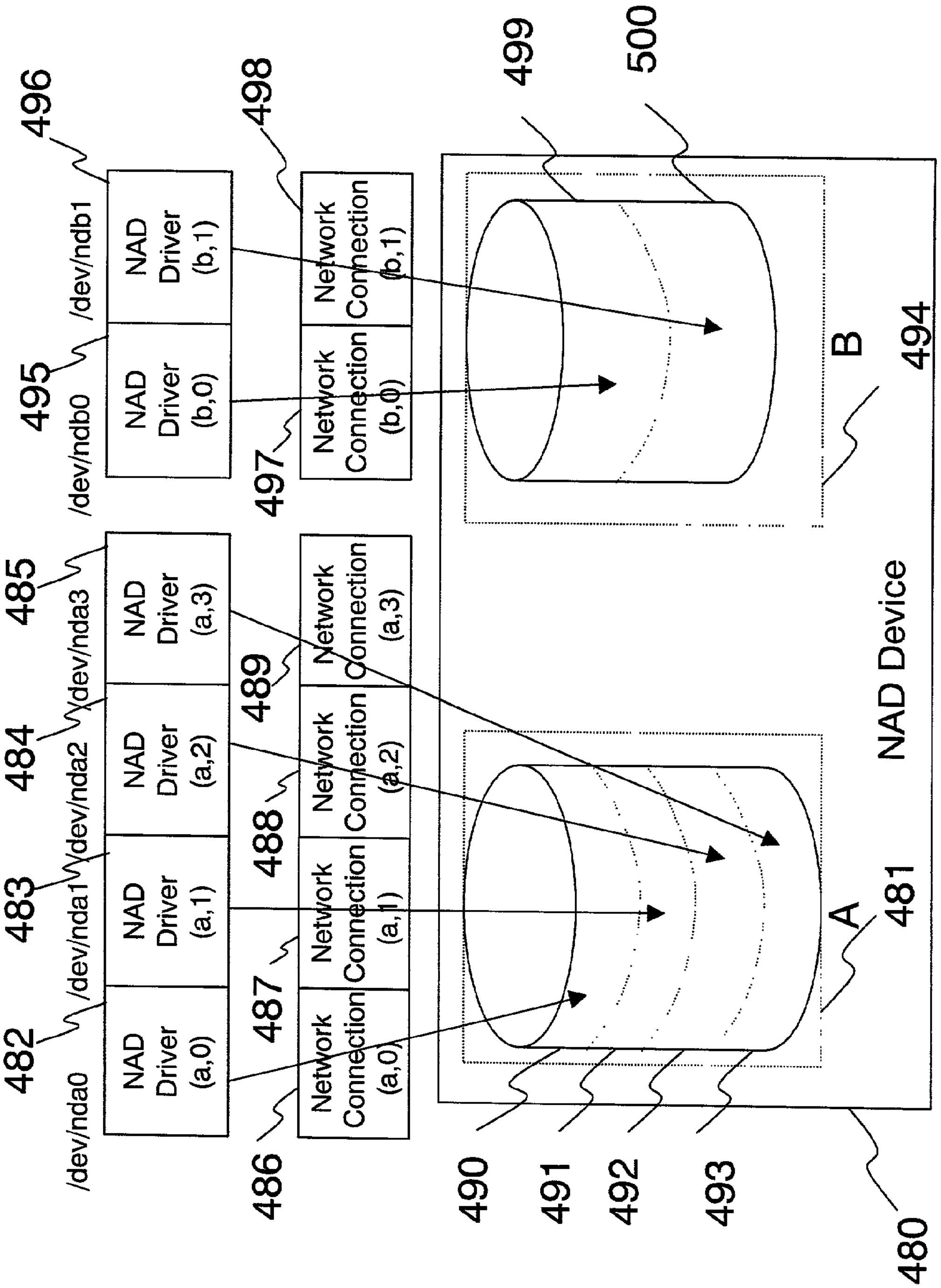

/dev/nda0
/dev/nda1
/dev/nda2
/dev/nda3
/dev/ndb0
/dev/ndb1
NAD Driver (a,0)
NAD Driver (a,1)
NAD Driver (a,2)
NAD Driver (a,3)
NAD Driver (b,0)
NAD Driver (b,1)
Network Connection (a,0)
Network Connection (a,1)
Network Connection (a,2)
Network Connection (a,3)
Network Connection (b,0)
Network Connection (b,1)
A
B
NAD Device
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500

FIG. 18

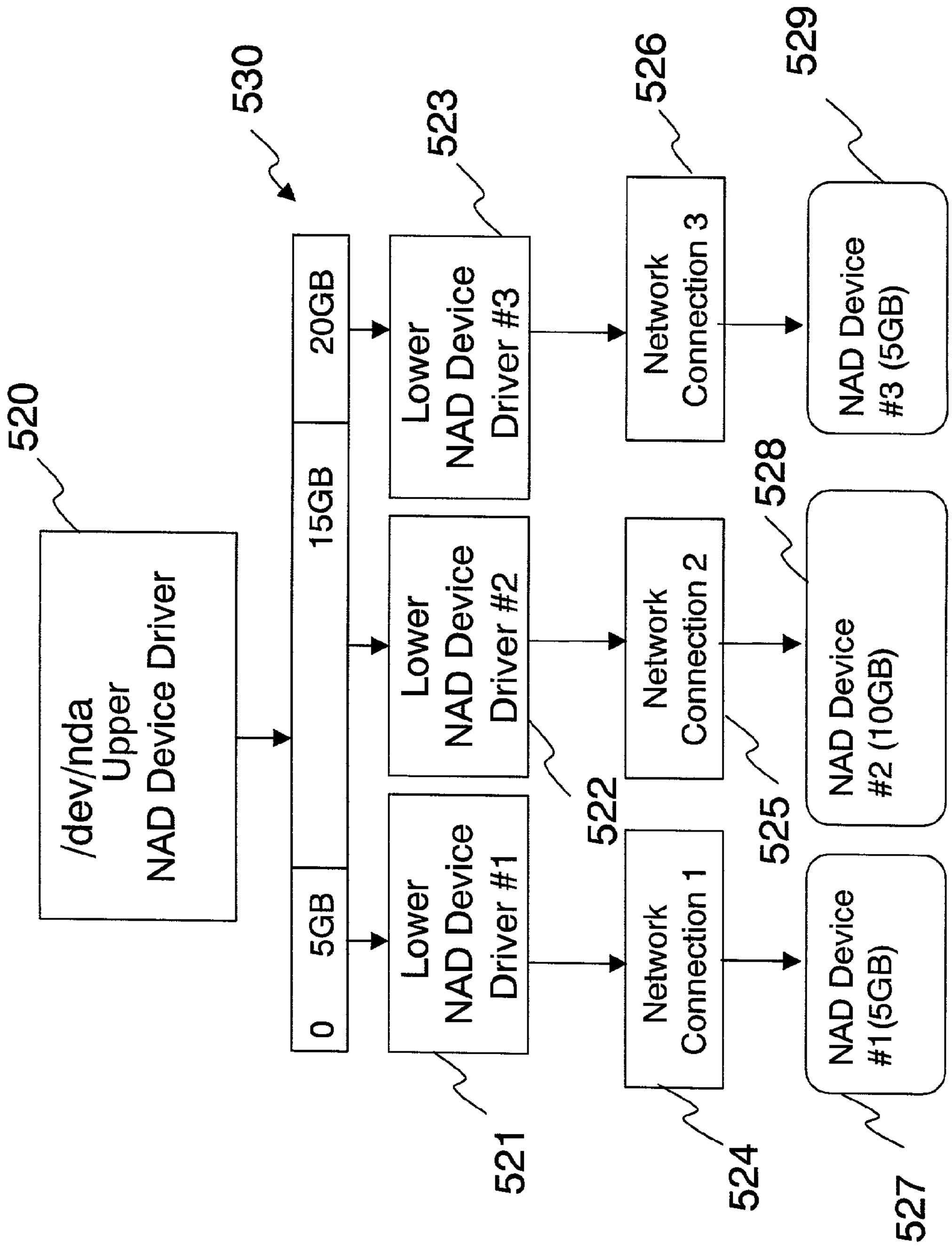

520
/dev/nda
Upper
NAD Device Driver
530
0
5GB
15GB
20GB
Lower
NAD Device
Driver #1
Lower
NAD Device
Driver #2
Lower
NAD Device
Driver #3
521
522
523
Network
Connection 1
Network
Connection 2
Network
Connection 3
524
525
526
NAD Device
#1(5GB)
NAD Device
#2 (10GB)
NAD Device
#3 (5GB)
527
528
529

FIG. 19

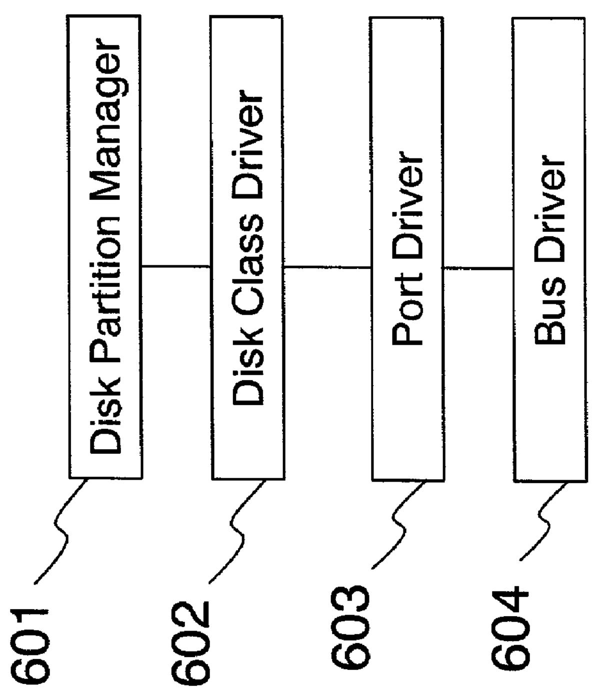


FIG. 20A

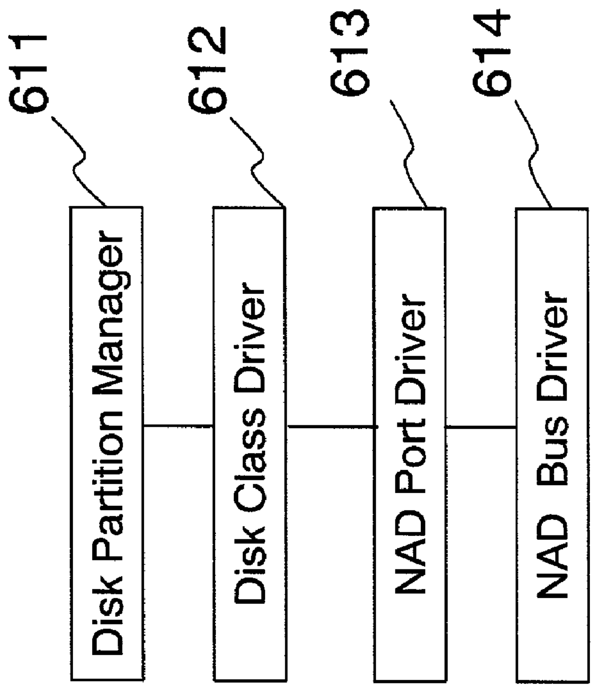


FIG. 20B

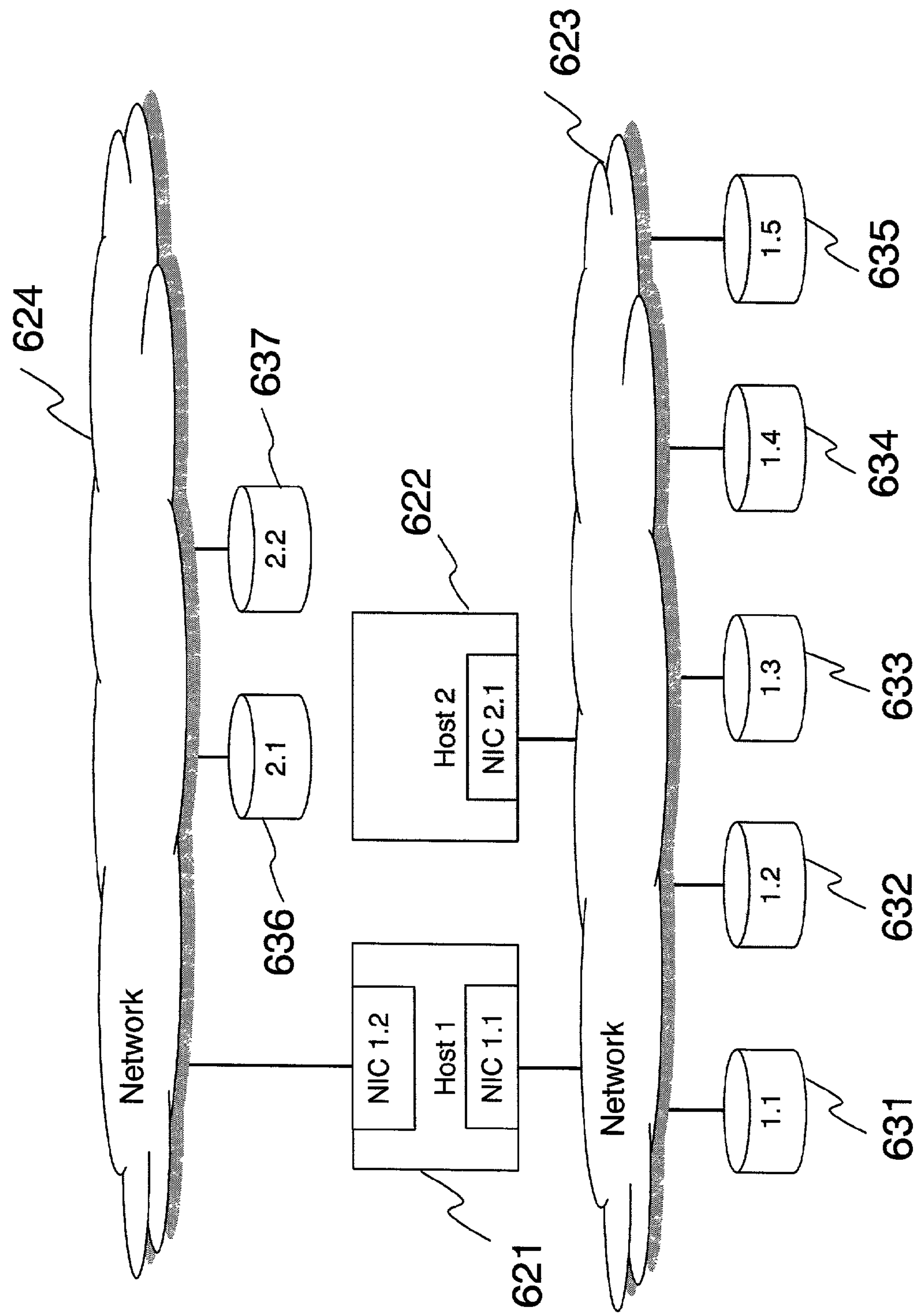

624
623
637
622
635
634
633
632
631
636
621
Network
Network
Host 1
NIC 1.2
NIC 1.1
Host 2
NIC 2.1
2.1
2.2
1.1
1.2
1.3
1.4
1.5

FIG. 21

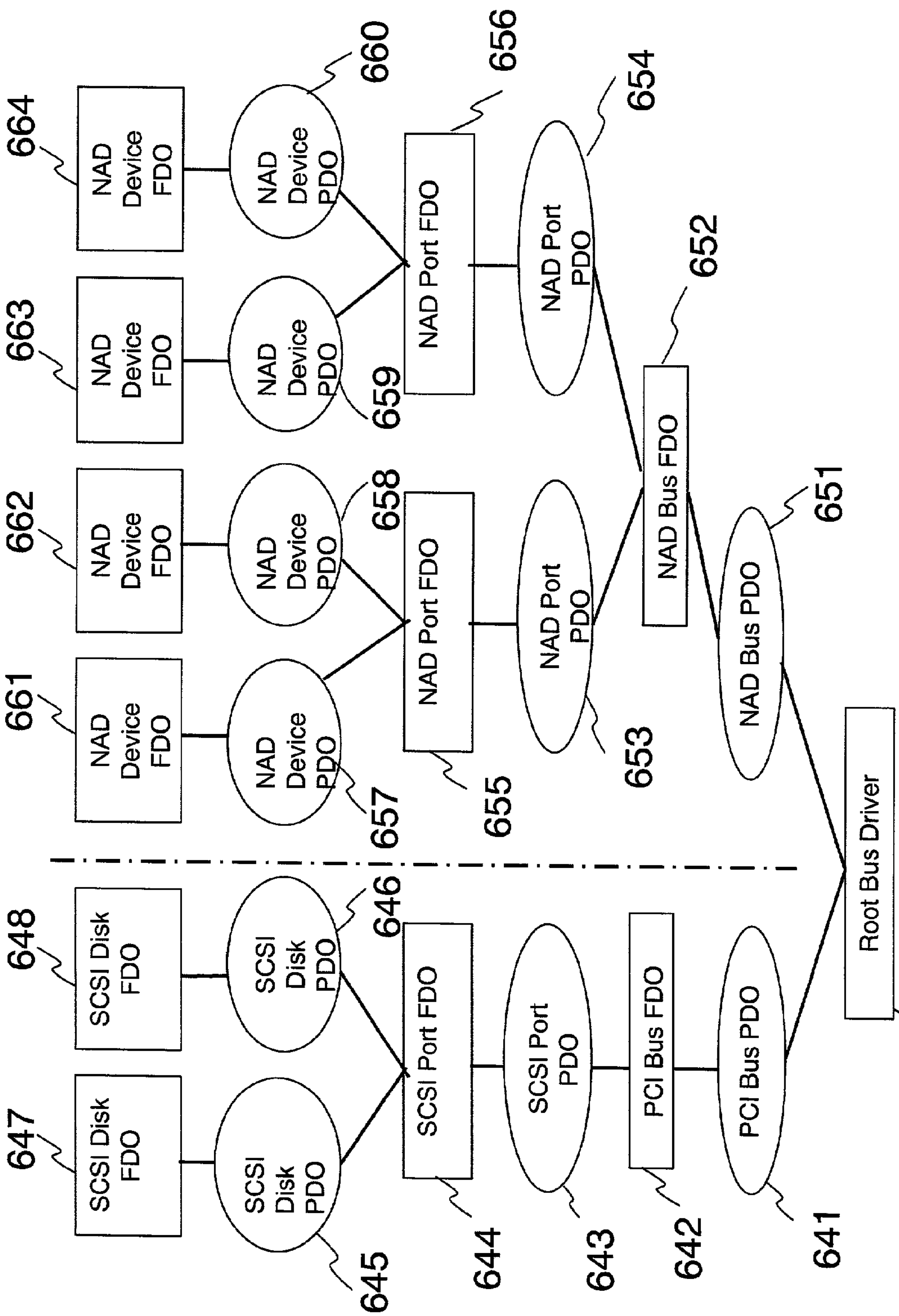

SCSI Disk FDO
SCSI Disk FDO
SCSI Disk PDO
SCSI Disk PDO
SCSI Port FDO
SCSI Port PDO
PCI Bus FDO
PCI Bus PDO
Root Bus Driver
NAD Device FDO
NAD Device FDO
NAD Device FDO
NAD Device FDO
NAD Device PDO
NAD Device PDO
NAD Device PDO
NAD Device PDO
NAD Port FDO
NAD Port FDO
NAD Port PDO
NAD Port PDO
NAD Bus FDO
NAD Bus PDO
640
641
642
643
644
645
646
647
648
651
652
653
654
655
656
657
658
659
660
661
662
663
664

FIG. 22

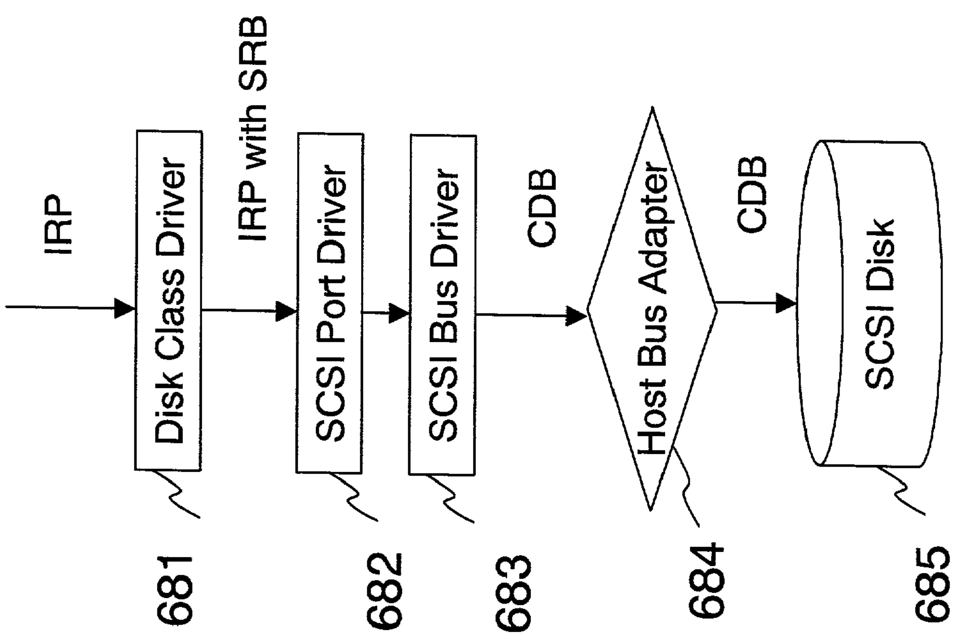


FIG. 23A

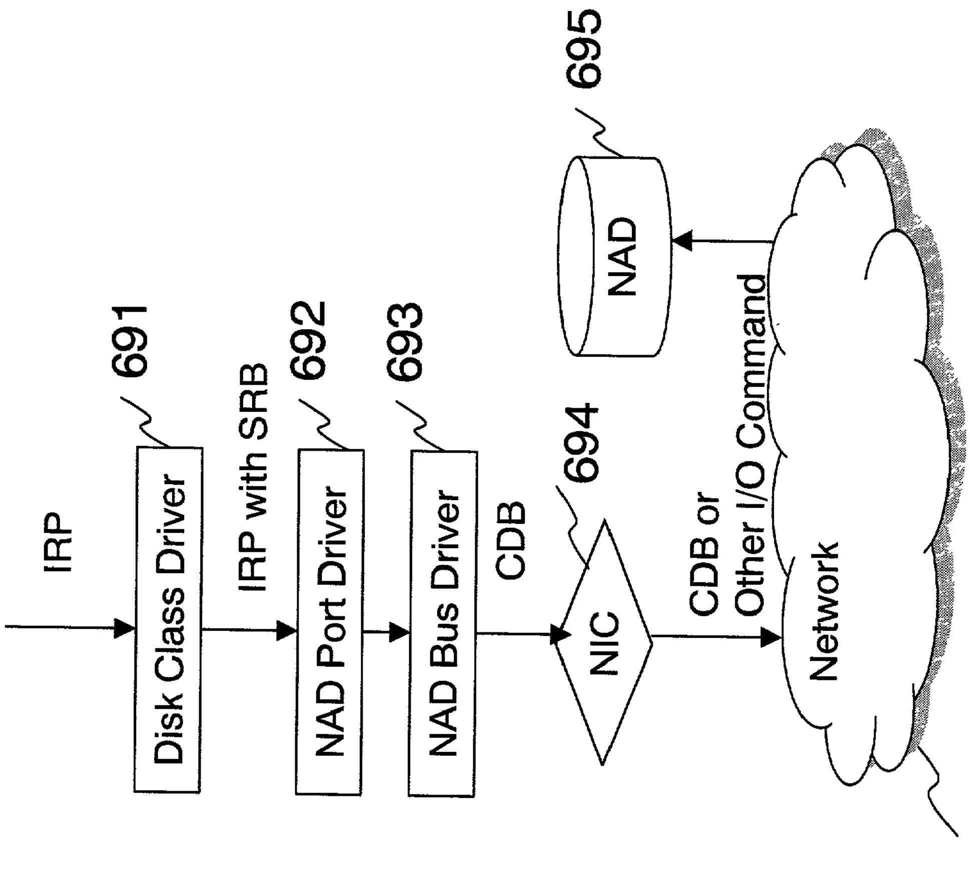


FIG. 23B

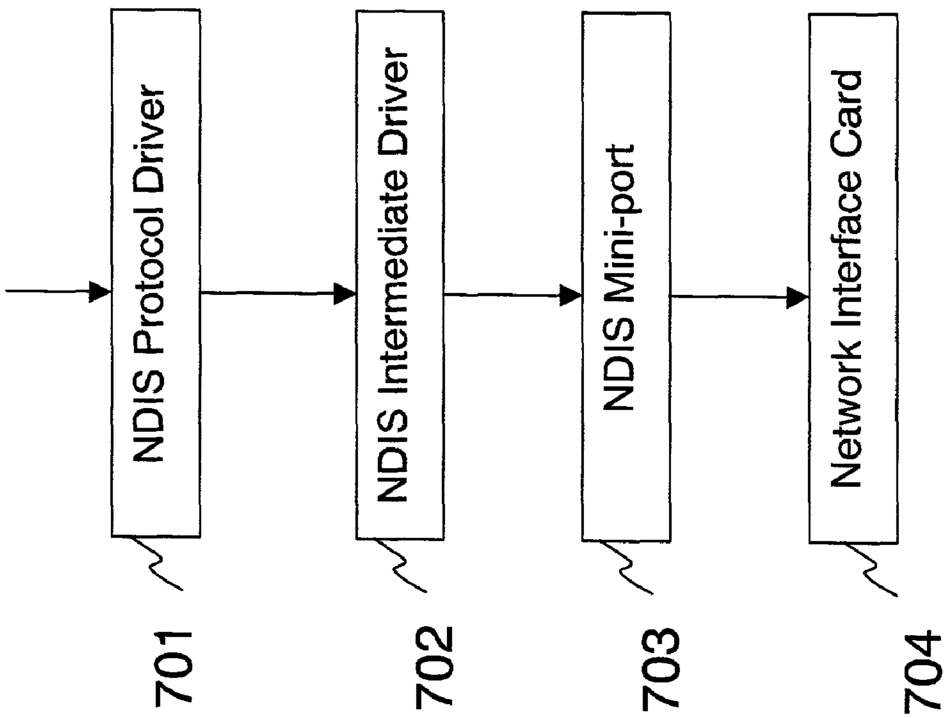

NDIS Protocol Driver
NDIS Intermediate Driver
NDIS Mini-port
Network Interface Card
701
702
703
704

FIG. 24

DISK SYSTEM ADAPTED TO BE DIRECTLY ATTACHED TO NETWORK

RELATED APPLICATION

This application claims the benefit of co-pending U.S. Provisional Application Ser. No. 60/240,344, filed Oct. 13, 2000, entitled "Disk System Adapted to Be Directly Attached to Network."

BACKGROUND OF THE INVENTION

1. Technical Field

This invention in general relates to computer systems. More specifically, this invention relates to a disk system or interface that can be directly attached to a network.

2. Description of the Related Art

As the Internet becomes popular, the amount of data that needs to be stored has drastically increased. Especially, there is a high demand for a high-capacity disk drive to store multimedia data. For example, a demand for a disk system having a disk capacity of terabytes per server is not unusual.

A tape drive or a CD drive may be used to store such amount of data, but its performance and user convenience are not matched to those of a hard disk drive. However, increasing the capacity of a hard disk in a conventional server system presents some problems.

There are NAS (Network Attached Storage) products that can be connected to a network, usually Ethernet, to provide a pre-configured disk capacity along with integrated system/storage management using the NFS (Network File System) protocol, the CIFS (Common Internet File System) protocol, or both on top of the IP protocol used on the Internet. The primary purpose of these protocols is to exchange files between independently operating computers. Therefore, the client using the NAS for file access experiences the difference between its local storage and the storage in the NAS systems.

The NAS is basically a stripped-down version of a file server having mainly the functions of storing and retrieving files. Accordingly, increasing a disk capacity using a NAS product amounts to adding a separate file server in practice, which presents many shortcomings. Since an NAS disk is not seen as a local disk to the client, the installation, movement, and administration of an NAS disk should be done only through the operating system and software offered as part of the NAS system. An NAS disk is installed in the inside bus of the NAS system, leading to a limitation to the number of disks that can be installed. Since the NAS system has a hard disk under its own operating system, the client cannot use an arbitrary file system to access the hard disk. Further, the NAS system requires an IP address. Overall, not only the installation and administration costs per disk are more expensive than those of a local disk, but also user convenience is severely restricted.

There is SAN (Storage Area Network) that uses the Fibre Channel technology. To use the devices connected to a SAN, a special-type of switch is needed. For example, Fiber Channel uses a Fibre Channel hub or a Fibre Channel switch. The SAN has some shortcomings. Typically, a separate file server is used. In general, the SAN equipment is expensive, and so is the administration cost of the SAN system because, for example, it often requires an administrator with a specialized knowledge on the system.

Therefore, there is a need for an interface that allows a disk system to be directly attached to a network, while still being

accessed like a local disk without the need of adding an additional file server or special equipment.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a disk system that can be directly attached to a network connecting to a host without going through a network file system.

Another object is to provide a disk system that can be recognized and used as a local disk to a host without requiring additional facility such as an additional file server, a special switch, or even an IP address, if appropriate.

Still another object of the present invention is to provide a disk system that can be conveniently connected to a server without much intervention of network/server administration.

Yet another object is to provide a low-cost disk system, many of which can be plugged into existing network ports to readily satisfy a disk capacity demand.

Further object is to provide an interface that allows a device attachable to a bus to be plugged into a network port.

The foregoing and other objects are accomplished by providing a network-attached disk (NAD) system that includes an NAD device for receiving a disk access command from a host through a network, a device driver at the host for controlling the NAD device through the network, where the device driver recognizes the NAD device as a local device. The host may run the UNIX or Windows family of operating systems. The NAD device includes a disk for storing data, a disk controller for controlling the disk, and a network adapter for receiving a disk access command from the host through a network port.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of an environment where a network-attached disk (NAD) system of the present invention is used.

FIG. **2** is an illustration of how multiple NAD devices may be accessed by multiple hosts through a network.

FIG. **3** is an example of how multiple NAD devices are accessed by multiple hosts.

FIG. **4** is another example of how each disk inside an NAD may be treated as a separate disk.

FIG. **5** is an illustration of how a block device driver, specifically an NAD device driver, is registered and unregistered under the UNIX operating system.

FIG. **6** is an illustration of the relation among the directory, device file, device driver, and device.

FIG. **7** is an illustration of how a request function directly issues a command to a device.

FIG. **8** is an illustration of how a request function activates a device accessing thread.

FIG. **9** is a block diagram of a local disk system and that of an NAD device running under UNIX.

FIG. **10** is an illustration of a device searching thread for identifying the attached NAD devices and for providing the NAD information to the NAD device management program.

FIGS. **11**A and **11**B are examples of network connections made between an NAD device driver and its corresponding NAD device using a connection setting thread.

FIG. **12** is an illustration of a method of implementing an NAD device driver, using a device accessing thread.

FIG. **13** is an illustration of a method of implementing an NAD device driver, without using a device accessing thread.

FIG. **14** is an example of an NAD device construction.

FIG. **15** is a functional block diagram of an NAD controller.

FIG. **16** is a simplified state transition diagram of a state machine used by the main controller of an NAD controller.

FIG. **17** is an illustration of how a disk inside an NAD device may be divided into disk partitions to which a device driver is assigned.

FIG. **18** is an illustration of how separate NAD device drivers may be generated so that a physically single disk can be assessed by different file systems.

FIG. **19** is an illustration of how the NAD system can recognize physically separate, several NAD disks as a logically single disk.

FIGS. **20**A and **20**B are illustrations of the hierarchies of the disk driver layers in the conventional disk system and the NAD system under the Windows 2000 operating system.

FIG. **21** is an illustration of a network environment where the NAD system of the present invention is used in the Windows 2000 operating system.

FIG. **22** is an example of a device stack created in the Windows 2000 operating system.

FIG. **23**A is an illustration of the flow of IRP, SRB, and CDB in a conventional disk system in the Windows 2000 operating system.

FIG. **23**B is an illustration of the flow of IRP, SRB and CDB in an NAD system in the Windows 2000 operating system.

FIG. **24** is an illustration of NDIS (Network Device Interface Specification) in the Windows 2000 operating system.

DETAILED DESCRIPTION OF THE INVENTION

FIG. **1** shows an environment where the present invention is used. A host **100** has a file system **101**, which may contain a local disk device driver **102** that controls a local disk **104** connected to an internal system bus **103**. A local device is defined as a device inside a standard-alone system as opposed to a network device connected to a network. Local devices are directly connected to a system bus often through an adapter called a host bus adapter allowing the host to communicate with the devices without going through any network, whereas network devices are not directly connected to a system bus, rather connected through an interface called a network interface card (NIC) installed on system bus. The local disk **104** may be a conventional IDE (Integrated Drive Electronics) disk or SCSI (Small Computer System Interface) disk.

The file system **101** also contains a network-attached disk (NAD) device driver **105** of the present invention that controls an NAD device **108** connected through a network adapter device driver **106** and a network **107** such as Ethernet. The NAD device **108** of the present invention contains one or more disks **109**. The network **107** is an existing general-purpose network for carrying storage traffic as well as other application traffic. This so called "front-end" network for carrying general-purpose network traffic is distinguished from a "back-end" network dedicated to storage such as that used in the conventional Storage Area Network (SAN) scheme.

The present invention features two main components: one is the NAD device driver **105** at the host and the other is the NAD device **108** attached to the network.

FIG. **2** shows an example of how multiple NAD devices are accessed by multiple hosts through a network. NAD device #1 **123** with disk(1,1) **126** and NAD device #2 **124** with disk(2,1) **127** and disk(2,2) **128** are accessed by Host #1 **120** through a network **122**, while NAD device #3 **125** with disk (3,1) **129**, disk(3,2) **130**, disk(3,3) **131** is accessed by Host #2 **121** through the same network **122**.

Each disk appears to the host as if it is a local disk to connected to the system bus of the host so that each disk can be dynamically installed or removed. The present invention achieves this by creating a virtual host bus adapter in purely software means that recognizes an NAD device as if it is connected to the system bus although there is no physical host bus adapter connected the NAD. This is distinguished from the conventional Network Area Storage (NAS) scheme where a NAS device connected through the NIC is still recognized as an independent file server connected to a network.

The Open Systems Interface (OSI) model defines 7 layers of protocols: a physical layer for electrical interface definitions, a data link layer for communication using data frames, a network layer for routing packets from one end to another, a transport layer for dividing messages into packets, a session layer for establishing communication session, a presentation layer for data presentation format, and an application layer for application programs. The present invention uses a data link layer protocol to contain storage commands into data link frames. Because the NAD device is not acting as an independent devices to the host, there is no need to use a network address such as IP address.

Since the specific configuration of the hosts and the disk systems can be dynamically changed, user convenience and portability is preserved as in the case of using a local disk. There is virtually no restriction to the number of disk systems that can be attached to the network, thus providing an unlimited disk storage capacity for a host.

FIG. **3** shows another example of how multiple NAD devices are accessed by multiple hosts through a network. NAD device #1 **143**, NAD device #2 **145**, and NAD device #5 **147** are accessed by Host #1 **140** through a network **142**, while NAD device #2 **144** and NAD device #4 **146** are accessed by Host #2 **141** through the same network **142**.

The disks contained in an NAD may be treated as separate disks so that each of them can be independently accessed by a host. FIG. **4** shows an example of treating each disk inside an NAD device as separate disks. Disk(1,1) **166** inside NAD device #1 **163**, disk(2,2) **168** inside NAD device #2, and disk(3,2) **170** inside NAD device #2 are accessed by Host #1 **160** through a network **122** while disk(2,1) **167** inside NAD device #2 **164** and disk (3,1) **169** and disk (3,3) **171** inside NAD device #3 **165** are accessed by Host #2 **161** through the same network **162**. Note that disk (2,1) **167** and disk (2,2) **168**, inside NAD device #2 **164**, are independently accessed by Host #1 **160** and Host #2 **161** respectively.

Block Device Driver

An embodiment of the NAD system will be explained with an example running the UNIX family of operating systems although other operating systems such as Windows may also be used.

Each block device for block data storage, such as a disk device, is assigned a major device number to distinguish among different kinds of block devices, and a minor device number to distinguish among same kinds of block devices. In UNIX, each device is accessed through a device file, which provides an interface for accessing the real device. Device files are usually generated in advance, each with a major device number and a minor device number as well as information on a block device driver.

The purpose of the device driver is to handle the requests made by the kernel with respect to a device. The device driver isolates device-specific codes to provide a consistent interface for the kernel. In order to activate the operation of a device driver, a device file and device driver routines must be prepared, after which the functions of the driver routines must be registered so that the operating system such as UNIX can understand their availability. This is usually done by passing the major number assigned to the device and the functions of the driver routines as parameters.

Registration and Unregistration of Block Device Driver

Once a block device driver is registered by passing the device's major device number and the driver functions as parameters, it may be unregistered by passing the major number.

Table 1 lists the functions used to either register or deregister a device driver.

TABLE 1

Functions	Description
Register_blkdev( )	- register a driver by taking a major number and driver functions as parameters
Unregister_blkdev( )	- unregister a driver by taking a major number

TABLE 2

TABLE 2 lists the general functions used by the local driver and the NDA driver.

Driver Function	Description
Read( )	- used to read data in the device
write( )	- used to write data in the device
ioctl( )	- used to change a particular value of a structure for a driver or to control input/output with respect to a device
open( )	- used to initialize a driver
release( )	- used to eliminate a driver
fsync( )	- used to reflect the content of buffer cache to the real device driver
check_media_change( )	- used to sense a change in the device condition
revalidate( )	- used to update device managed by the deriver and device driver itself

Table 3 lists examples of the driver functions specific to the IDE local disk driver and the NAD driver.

Use of Block Device

Once a block device driver is registered and its device file is generated, read/write is done to the device file to access the real device. The device file, however, is not directly called by the user, rather called after being mounted to the file system. Before being mounted, each block device file must be formatted according to a particular file system. Since the NAD device driver of the present invention is prepared in the same way as a conventional local disk driver, the set of I/O commands used to format a conventional local disk can also be used to format a disk in the NAD device. In addition, since NAD devices are controlled in the device driver level, they can be formatted in a required format independent of the file system used.

FIG. **6** shows the relation among the directory, device file, device driver, and device. The left side shows attachment of a conventional local disk system where a device file **201** mounted on a directory **200** is used by a local disk device driver **202** to control a local disk **203**. The right side shows an NAD system of the present invention where a device file **204** mounted on the directory **200** is used by an NAD device driver **205** to control a NAD device **207** through a local area network (LAN) **206** such as Ethernet. The two relations are similar except that the NAD device is accessed through the network.

Structure of Block Device Driver

Each block device driver has an I/O request queue to store the I/O requests to the device. The stored requests may be re-scheduled for the purpose of improving the performance. Besides the I/O request queue, each block device driver needs a request function to process the I/O requests in the queue.

FIG. **7** shows a situation where the request function directly issues a command to a block device. An NAD device driver **220**, using a device **222** and a file system **223**, has a queue **224** that stores I/O requests **225** through **228**. The NAD device driver **220** has a request function **229** that issues a command to the NAD device **221** by taking a currently processed request **225**.

TABLE 3

Driver Function	IDE Local Driver Function	NAD Driver Function
read( )	ide_read( )	netdisk_read( )
write( )	ide_write( )	netdisk_write( )
ioctl( )	ide_ioctl( )	netdisk_ioctl( )
open( )	ide_open( )	netdisk_open( )
Release( )	ide_release( )	netdisk_release( )
fsync( )	ide_fsync( )	netdisk_fsync( )
check_media_change( )	ide_check_media_change( )	netdisk_check_media_change( )
revalidate( )	ide_revalidate( )	netdisk_ide_revalidate( )

FIG. **5** shows an example where a block device driver, specifically an NAD device driver, is registered and unregistered. Initially, an IDE device driver **181** with major device #3 is created as well as some null device drivers such as major device #0 **180** and major device #n **182**. Major device #60, **183**, the NAD device driver that is assigned a major device number of 60, is registered by using a device registration function of register_blkdev(60, fops) **185**. Later the NAD device driver is de-registered into major device #60 **184**, a null device driver, by using a device de-registration function of unregister_blkdev(60) **186**. The figure shows that NAD device driver of the present invention is installed in the same way as the existing block device drivers.

FIG. **8** shows a situation where the request function **229**, instead of directly issuing a command, activates a device accessing thread **230** so that the device accessing thread **230** can issue a command based on the information in the request queue. A thread refers to a single use of a program that can handle multiple users.

Constitution of Local Disk System and NAD System

FIG. **9** shows the constitution of a conventional local disk system and that of the NAD system operating under UNIX. Under a file system **260**, a conventional local disk **264** attached to a local bus **263** is accessed by a conventional local disk device driver **261** through a bus interface **262**. Under the same file system **260**, an NAD device **272** with a disk **273**, attached to a network **271**, is accessed by an NAD device driver **265** through a network interface including a network protocol stack **266**, a network adapter device driver **267**, a bus interface **268**, and a network adapter **270**.

Since an NAD device is to be used like a local disk, the conventional local disk system and an NAD system of the present invention share a basic structure. The difference is that since an NAD system must communicate with an NAD device through a network, a protocol stack is added for network communication. The NAD driver delivers an I/O command to an NAD device through a network adapter and receives a response from the NAD device.

When an NAD device is accessed, either DMA (Direct Memory Access) or PIO (Programmed Input/Output) may be used. A conventional disk device driver operates in a DMA mode by issuing a DMA I/O command to a local disk with a starting buffer address and a byte transfer count. The local disk then takes over the data transfer, after which it interrupts the CPU. Similarly, the NAD device driver may be implemented to operate in a DMA mode by having the NAD device driver deliver an I/O command to an NAD device, which then completes the data transfer, after which it interrupts the CPU.

The conventional disk driver operates in a PIO mode by the CPU transferring data directly through data registers of the disk device until a particular data block is processed. Similarly, the NAD driver may be implemented to operate in a PIO mode by having the NAD device driver deliver a command to an NAD device and continue to transmit/receive data until a particular block of data is processed.

The network protocol that can be used in the present invention is not restricted to a particular protocol. It can be any connection-oriented protocol including TCP/IP. A connection-oriented protocol ensures that packets are not lost and packets are received in the order they are transmitted. If TCP/IP is used, an IP address must be used for each NAD device.

Local Disk Driver and Generation of NAD Driver

Once UNIX starts, if hardware scan detects any conventional local disks, their corresponding drivers are generated according to the units of the local disks or according to the units of disk partitions. In a similar fashion, NAD devices are identified during initial hardware scan and their corresponding drivers acting as a virtual host bus adapter must be generated. The drivers may be generated automatically by using a device searching thread that periodically identifies NAD devices attached to the network or manually by a system administrator using an NAD management program.

FIG. **10** shows a device searching thread for identifying the attached NAD devices and for providing the NAD device information to an NAD device management program. A thread **280** is run in a host **290** through a network protocol stack **282** and a network adapter driver **283** to identify NAD devices **285** through **289** together with the size and device file of each NAD to provide the information **281** to NAD device management program. Once informed of NAD device files available, the user then mounts a selected NAD device file to use a particular NAD device as a local disk.

Network Connection between NAD Device Driver and NAD Device

In a conventional local disk, disk I/O is performed by reading/writing to I/O ports of the disk controller attached to the internal system bus. But the NAD device driver performs I/O to the corresponding NAD device through a network link. Instead of read/write to an I/O port, I/O is performed by read/write to a network connection such as a socket in UNIX. Therefore, a network connection such as a UNIX socket must be set up between the NAD device driver and NAD device.

FIGS. **11**A and **11**B show examples of network connections between an NAD device driver and the corresponding NAD device using a connection setting thread. NAD device #1 **302** is connected to NAD device driver #1 **301** through a network connection #1 **303** created by the ioctl( ) function, **304** while NAD device #2 **307** is connected to NAD device driver #2 **306** through a network connection #2 **308** created by the ioctl( ) function **309**.

Implementation of NAD Driver

FIG. **12** and FIG. **13** show two methods of implementing an NAD device driver, the former with a device accessing thread, and the latter without a device accessing thread.

FIG. **12** shows three NAD drivers **320**, **321** and **322** with the device files of "/dev/nd0", "/dev/nd1", "/dev/nd2" to access NAD device #1 **323**, NAD device #2 **324**, and NAD device #3 **325**, respectively. Each device file is mounted to "/tmp", "/var", "/" directory in the file system **326**, respectively. User threads **327**, **328** and **329** for accessing the file may read/write on the NAD device through the file system **326**. A connection setting thread **331** provides the list of NAD devices available to an NAD device management program **330**. Based on the user's input, the connection setting thread **331** creates network connections **332**, **333** and **334**, as necessary.

Referring to FIG. **12**, when the user thread requests a file through a file system, the file system first checks the buffer cache to find out whether the requested file block is in the buffer. If the block is in the buffer, the user thread refers to the block. But if the block is not in the buffer, data must be read from the NAD device. The user thread puts the request on the request queue, activates an NAD accessing thread **335** (or **336**, **337**) responsible for NAD device control through a request function, and the user thread blocks itself. The user thread blocked is awakened later by the NAD accessing thread, such as **335**, that received the corresponding data.

FIG. **13** is similar to FIG. **12** except that the user thread now directly requests data from the NAD device rather than using an NAD accessing thread. For example, the user thread puts the request on the request queue, activates a software interrupt that will actually handle block data transfer between the NAD device and the host, and the user thread blocks itself. Once the data transfer is done, an interrupt is generated to wake up the blocked thread.

Communication Protocol between Host and NAD Device

When a host NAD device driver accesses an NAD device for I/O, the position of the first block and the number of blocks are given as parameters of the I/O command. Or, in the case of SCSI, the I/O command may be in the form of a CDB (Command Descriptor Block).

To transfer the CDB or the block transfer information, a reliable communication protocol is necessary. The present invention is not limited to a particular kind of communication protocol as long as a connection-oriented protocol is used including TCP/IP. A connection-oriented protocol means that packets can be retransmitted in the case packets are lost, and received packets are arranged at the receiver end in the order they were sent.

NAD Device

FIG. **14** shows a functional block diagram of the NAD device of the present invention. A preferred embodiment of the NAD device is comprised of an NAD controller **401** for controlling the whole NAD device, memory **402**, a disk controller **403** for executing a disk access command, one or more disks **405**, **406**, and a LAN adapter **403** for receiving a disk access command from a host through a network. The NAD controller **401**, the memory **402**, the disk controller **403**, and the LAN adapter **404** are all connected to a bus **419** internal to the NAD device.

The disk controller **403** is a module that performs disk I/O operations by controlling the disks **405** and **406** over internal disk channels. The disk controller **403** is further comprised of one or more disk channels **407** and **408** controlled by a channel controller **409**, a buffer manager **410**, some registers **411**, and a bus interface **412**. The buffer manager **410** consults the registers **411** to obtain a disk sector number and a channel to execute a disk access command. The buffer manager **410** also commands the channel controller **409** to transfer data from the memory to disk channel **407** or **408** or vice versa as a result of executing a disk access command. The channel controller **409** actually accesses the disk over the disk channel **407**, **408** to transfer data from the disk to the memory or vice versa.

The LAN adapter **404** is a module that receives disk I/O command packets from the host and transmits replay packets over the network. The LAN adapter **404** is further comprised of a physical network interface **413** for interfacing with the network, a MAC (media access control) controller **414**, transmit buffer **415** for storing transmit data, a receive buffer **416** for storing receiving data **416**, registers **417**, and a bus interface **418**.

The bus interface **418** transfers data from the bus to the transmit buffer **415**, the receive buffer **416**, and the registers **417**, or vice versa. The MAC controller **414** transfers data to the physical network interface **413** so that the physical network interface can transmit the data to the host computer. When the physical network interface **413** receives a disk I/O request packet from the host computer, it transfers the packet to the MAC controller **414** so that the MAC controller can extract necessary data from the packet and transfer the data to the receive buffer **416**.

FIG. **15** shows that the NAD controller **401** may be comprised of a main control **420** for controlling the NAD, a buffer management module for caching data in the disk **421**, a memory management module for managing assignment of memory space **422**, a disk controller driver **423** for interfacing with the disk controller, a network adapter driver **424** for interfacing with the network adapter, and a bus interface **425** for interfacing with the bus inside the NAD.

The NAD controller **401** mainly executes I/O commands from the host's NAD device driver, but it can perform other additional functions. For example, a filter program can be installed to NAD so as to provide features that are not offered in the host, for example, a back up operation. Other examples include access control, access share, access right transfer, etc. Specifically, a filter program can be installed to limit access to an NAD device to a certain time period, to allow several hosts simultaneously access an single NAD, or to transfer one host's access rights to another host. The NAD device driver at the host can request to execute the filter program at the time of I/O command execution through the ioctl( ) function in UNIX.

FIG. **16** shows a simplified state transition diagram of a state machine used by the main controller **420**. At the 'init' state **440**, the state machine initializes all the NAD hardware and allocates memory for the disk controller **403** and the LAN adapter **404**. Upon completing the initialization process, the state machine makes a transition to 'wait-command' state **441** where the NAD system waits for an incoming I/O command from the host computer over the network. When such I/O command is received from the host computer, the state becomes the 'disk_access' state **442** where an appropriate disk I/O operation is performed through the disk controller. Upon completing the disk I/O, the state moves to the 'transmit_reply' state **443** where the NAD device sends the result to the host computer through the LAN adapter **404**. A person skilled in the art would appreciate that the state machine can be readily realized with a CPU and memory.

Network Adapter Driver and Disk Controller Driver

The network adapter and the disk control driver can be implemented at least in two ways. One uses an interrupt mechanism through DMA (Direct Memory Access) and the other uses polling through PIO (Programmable I/O). The former has the advantage of easy programming so that other jobs can be executed without a complete disposition of disk controller data. The latter has the advantage of dispensing with time delay due to interrupts, but has the disadvantage of an inefficient processor usage due to the time spent for continuous read and write.

Additional Embodiments of NAD Drivers

Usually, an NAD device driver is generated for each disk unit attached. However, just as a local disk may be partitioned, the disk inside an NAD device may also be partitioned into several disk partitions where each disk partition can be accessed by a separate device driver. Alternatively, several disks located in the physically separate NAD devices may be combined for use as a logically single disk.

FIG. **17** shows an example where the disk inside an NAD device may be divided into several disk partitions where all of the partitions are assigned a single device driver. An NAD driver A **462**, for example, is assigned to four partitions **463-466** so that the NAD driver A **462** refers the partition table in order to handle I/O requests directed to specific partitions **468** through **471** of a disk **461** inside a NAD device **460**, respectively, using a same network connection **467**. Similarly, an NAD driver B **473** is assigned to two partitions **474** and **475** so that the NAD driver B **473** can be used to control two disk partitions **477** and **478** of a disk **472** inside the NAD device **460**.

FIG. **18** shows an example where separate NAD device drivers may be generated so that a physically single disk can be assessed by different file systems. Disk A **481** inside an NAD device **480** is divided into four partitions **490** through **493**, and four separate NAD driver(a,0) through driver (a,3) **482** through **485** are created so that each NAD driver can control each disk partition through a separate network connection **486** (**487**, **488**, or **489**). Similarly, disk B **494** inside the NAD device **480** is divided into two partitions **499** and **500**, and two separate NAD drivers(b,0) and NAD drive (b,1) **495** and **496** are created so that each NAD driver can control each disk partition through a separate network connection **497** (or **498**). Since different network connections are used, this configuration enables a physically single hard disk to be mounted to different file systems.

FIG. **19** shows an example of how the present invention can recognize physically separate, multiple disks in different NAD devices as a logically single disk. Specifically, FIG. **19** shows that three lower-level NAD device drivers **521**, **522** and **523** controlling NAD device #1 **527** of 5 GB, NAD device #2 **528** of 10 GB, and NAD device #3 **529** of 5 GB, respectively, through separate network connections **524**, **525** and **526**, are united into a single upper-level NAD driver **520** partitioned into a configuration **530**. The file system mounts "/dev/nda" to access the total space of 20 GB.

NAD System Running under Windows Operating System

The foregoing system and method explained using examples running under the UNIX family of operating systems can equally be applied to implement an NAD system running under the Windows™ family of operating systems so that it can be recognized as a local disk. For example, an NAD device may treated as a local disk per se by a Windows 2000™ host so that all disk operations exercised by the host control a local disk, including formatting and partitioning, can be done to the NAD device.

This feature differentiates the present invention from other solutions, such as those provided by the NAS technology, which expand disk space through the intervention of a file system instead of directly adding individual disks at the device level of the host system. At the same time, since the NAD device is to be accessed through the network, the present invention redirects the disk I/O request to the network interface otherwise would be directed to the disk controller connected to the inside system bus in the case of using conventional local disks.

In other words, the present invention creates a virtual host bus adapter in purely software means by modifying a driver at the host so that the host recognizes the NAD device as if it is connected to the system bus through a physical host adapter although actually there is no physical host adapter connected to the bus. Since an NAD device is accessed as if it is a local device connected to the internal bus of a host, there is no need to use network addresses such as IP addresses for the host to communicate with the NAD device. Instead, data link frames containing storage commands are exchanged between the host and the NAD device.

FIGS. **20**A and **20**B shows a comparison of the hierarchy of the disk driver between the conventional disk system and the NAD system of the present invention. FIG. **20**A shows conventional disk driver layers in Windows 2000, which are organized in a hierarchy comprising a disk partition manager **601**, a disk class driver **602**, a port driver **603**, and a bus driver **604**.

In the Windows 2000 operating system, the generic term, 'bus', refers to a piece of hardware to which devices connect electronically. Not only does it include things like the PCI bus, but it also includes anything that can have multiple devices plugged into it such as a SCSI adapter, a parallel port, a serial port, a USB hub, and so on. One responsibility of the bus driver is to enumerate devices attached to the bus and to create physical device objects for each of them as necessary in Windows 2000. Therefore, the bus driver is a collection of software routines that contain the information about the specific bus and the functions that allocate system resources such as port addresses and IRQ numbers to the devices connected to the bus. It is the port driver that contains routines required to perform most of the actual disk I/O operations.

The major feature of the present invention is to replace the conventional bus driver and the port driver with a new bus driver and a new port driver so that the NAD devices can be recognized as the same as the local disks and the disk I/O operations can be performed to the NAD devices through the network port of the Windows 2000 host.

FIG. **20**B shows the driver layers of the present invention, which have an NAD port driver **613** and an NAD bus driver **614** replacing the corresponding conventional Windows 2000 driver layers of FIG. **20**A. The NAD bus driver **614** implements a virtual host bus adapter, through which disk I/O operations are to be done to and from a set of NAD devices. The NAD port driver **613** implements a set of routines required to perform actual disk I/O operations by redirecting the I/O requests to the NAD devices through the network port of a Windows 2000 host.

FIG. **21** shows an example of a network environment where NAD devices of the present invention are attached to multiple hosts. The example shows that both Host #1 **621** and Host #2 **622** run Windows 2000 connected to Network #1 **623** and Network #2 **624**. Host #1 uses disk(1,1) **631** and disk(1,3) **633** through Network #1, disk (2,1) **636**, and disk(2,2) **637** through Network #2 **625**. Similarly, Host #2 uses disk(1,2) **632**, disk(1,4) **634** and disk(1,5) **635**.

Given the NAD bus driver and the NAD port driver, a Windows 2000 system creates device stacks as specified in Windows 2000 in order to be able to process I/O requests. Each device in Windows 2000 is expressed in terms of device objects organized in a stack structure. Device objects are data structures that the Windows 2000 system creates to help software manage hardware. Many of these data structures can exist for a single piece of physical hardware. The lowest-level device object in a stack is called a physical device object (PDO). Above a PDO in a device object stack is an object called a functional device object (FDO). There may be a collection of filter device objects below and above the FDO. The Plug and Play (PnP) Manager component of Windows 2000 constructs the stack of device objects at the command of device drivers. The various drivers that occupy the stack for a single piece of hardware perform different roles. The function driver manages the device, and the bus driver manages the connection between the device and the computer.

FIG. **22** shows an example of device stacks that may be created to implement the present invention, where all filter device objects are omitted for the simplicity. Shown on the left half is a layer of recursively enumerated SCSI devices on top of the PCI bus, which is typically the case when SCSI disks are connected to the host's PCI bus. In the first instance, a PnP Manager has a built-in driver for a virtual root bus that conceptually connects computer to all the hardware that can't electronically announce its presence including hardware bus such as PCI. The root bus driver **640** gets information about the PCI bus from the registry to create a PCI bus PDO **641**, a PDO for the PCI bus, where the registry was initialized by a Windows 2000 setup program.

Having created the PCI bus PDO **641**, the PnP Manager then loads functional drivers for the PCI bus, thus creating a PCI bus FDO **642**. The functional driver of the PCI bus can then enumerate its own hardware devices attached to the PCI bus, where the example system in FIG. **21** assumes to have a set of SCSI devices, to create a SCSI port PDO **643**. Once the SCSI port PDO **643** is created, the PnP Manager then loads drivers for SCSI port device, thus creating a SCSI port FDO **644**. Similarly, SCSI disk PDOs, such as **645** and **646**, are created for each of the individual SCSI disks on top of the SCSI port, and SCSI disk FDOs, such as **647** and **648**, are in turn created by loading the disk class driver.

Shown on the right half of FIG. **21** is the corresponding device stacks for the NAD devices that would be created by using the NAD bus driver and NAD port driver replacing the PCI bus driver and the SCSI port driver, respectively. On top of the root is a NAD BUS PDO **651**, the PDO of the NAD bus that is not conventional hardware bus such as PCI, but a bus required to fit in the Windows 2000 device stack in order to provide virtual bus for NAD devices. On top of the NAD bus PDO **651**, the PnP Manager creates a NAD bus FDO **652** by loading a NAD bus driver.

A set of NAD Port PDOs **653** and **654** for each of individual network interface cards (NICs) of the Windows 2000 host are then created recursively since one NAD port is implemented to correspond to one NIC of the host in the present invention. On top of each NAD port PDO such as **653** or **654**, each NAD port FDO such as **655** or **656** is created by loading a NAD port driver. It is the NAD port driver that performs the actual NAD disk I/O operations. The NAD port driver should handle the NAD device I/O requests by redirecting the I/O requests and obtaining the I/O replies to and from the corresponding NAD devices through the specific NIC. The NAD port FDO such as **655** or **656** then creates individual NAD device PDOs such as **657**, **658**, **659** or **660** on top of the specific NAD port for individual NAD devices that can be accessed through the specific NAD port bound to a specific NIC.

FIG. **22** shows that for the example in FIG. **21**, two stacks of NAD port objects **653** and **654** are created because Host #1 has two NICs. Host #1 also has four NAD device PDOs **657** through **660**, two for each NAD port, because NAD devices, i.e., disk(1,1) **631** and disk(1,3) **633** and disk(2,1) **636** and disk(2,2) **637** are to be accessed through the NIC(1,1) and NIC(1,2) respectively. For each individual NAD device PDO such as **657**, **758**, **659** or **660**, the PnP Manager loads disk class driver to create a NAD device FDO such as **661**, **662**, **663** or **664**.

Note here that the only NAD bus driver and NAD port driver are to replace the conventional bus driver and SCSI port driver respectively in order to substitute the NAD devices for the conventional local disks. Disk class driver and other higher level drivers of Windows 2000 should remain intact without a single change in order for the Windows 2000 system to recognize the NAD device as same as a local disk.

In Windows 2000, each request for an operation affecting a device uses an I/O request packet (IRP). IRPs are normally sent to the topmost driver of a stack for the device and can percolate down the stack to the other drivers. At each level, the driver decides what to do with the IRP. Sometimes, the driver does nothing except passing the IRP down. The driver may completely handles the IRP without passing it down or process the IRP and pass it down. In the case of disk I/O, for example as shown in FIG. **20**B, an IRP for a file I/O sent to the file system driver is passed to a volume manager, a disk class partition manager, to a partition manager, and to disk class driver.

It is the disk class driver where a SCSI Request Block (SRB) is created to be included in the IRP as necessary. An SRB is a data structure specified in the Windows 2000 for SCSI device I/O. If the IRP is for the conventional local disk, the disk class driver passes the new IRP down to a SCSI port driver that completes actual disk I/O operation. If the IRP is for the NAD device connected to the network, the disk class driver passes the IRP down to NAD port driver that completes NAD device I/O through the network interface.

Without regard to the particular device type of the disk, local or NAD device, it is the feature of the Windows 2000 device stack as shown in FIGS. **20**A and **20**B that an IRP for a specific disk, local or network-attached, is directed eventually to the corresponding disk. This is because separate disk object stacks are created for each of the individual disks. FIG. **22** shows that separate SCSI disk FDO/PDOs and NAD device FDO/PDOs are bound to each of the individual local disks and NAD devices, respectively.

The present invention replaces the conventional disk bus driver and port driver with the new NAD bus and port drivers as shown in FIG. **20**B so that NAD devices would be recognized as local disks by the Windows 2000 system.

All of the Windows 2000 device drivers have functions to create and remove the FDO for each device and dispatch functions to handle IRP passed down from the above driver layer. The major and minor function numbers in the IRP specify which of the dispatch functions will be invoked.

The following is an explanation of the actual software modules implemented in the NAD bus driver and port driver of the present invention in order to implement the NAD system for Windows 2000.

NAD Bus Driver

The NAD bus driver is a set of software modules that implement a virtual host bus adapter to which NAD ports are to be attached, where the individual NICs of a host are realized as NAD ports. The functions of the NAD bus driver are basically the same as those of a conventional PCI bus driver in Windows 2000. The NAD bus driver performs the functions of finding out the number of the NICs installed in the host computer and enumerating those NICs to create an NAD port PDO for each of the existing NICs. It also performs the functions of creating, starting, stopping, and removing an NAD port. In the NAD system, an individual NIC is regarded as an independent NAD port so that NAD disk ports for NAD devices are created according to the number of independently operating network units. See the example configuration shown FIG. **22**.

The difference between the NAD bus driver and a conventional PCI bus driver is that the NAD bus driver is for NAD devices that are physically separated from the system bus of the host but are connected through network ports. Unlike a conventional Windows 2000 system that detects plug-in of a device to or removal of a device from the hardware bus through a hardware interrupt, the NAD bus driver is implemented by creating a kernel thread to install and remove an NAD port on the NAD bus. The kernel thread created by the NAD bus driver starts to work when an IRP with IRP_MJ_PNP as its major function number and IRP_MN_START_DEVICE as its minor function number is sent to the NAD bus FDO from the PnP Manager. The thread terminates when the NAD bus FDO is removed. The thread periodically detects existence of NICs. If a new NIC is detected, the thread creates a new NAD port PDO for the NIC and includes the newly created NAD port PDO into its own list of NAD port PDOs. The thread then invokes the PnP Manager to have the NAD port PDO recognized by the system. Removal of an NIC is also detected by the thread since the thread can detect the absence of the NIC of which NAD port PDO previously created would be found in the above mentioned list without the corresponding NIC. If an NIC is found to have been removed from the host, the thread removes the corresponding NAD port PDO from its list and invokes the PnP Manager to remove the NAD port from the Windows 2000 system.

The software routines implemented in the NAD bus driver can be classified into five categories. The following tables list some of the routines implemented in the NAD bus driver with brief explanations.

TABLE 4

Basic functions	
DriverEntry( )	- executed when the driver is initially loaded - registers dispatch routines of the NAD bus driver - initializes the variables used by the driver
NADBusUnload( )	- recovers resources occupied by the driver when the driver is unloaded

TABLE 4-continued

Basic functions	
NADBusAddDevice( )	- creates NAD bus FDO - initializes the value of the NAD bus FDO

TABLE 5

Dispatch functions	
NADBusCreate( )	- processes the 'IRP__MJ__CREATE' IRP
NADBusClose( )	- processes the 'IRP__MJ__CLOSE' IRP
NADBusPnp( )	- processes the 'IRP__MJ__PNP' IRP - determines whether the IRP passed is to NAD bus FDO or to NAD port PDO, and invokes NADFDOPnP( ) or NADPDOPnP( ) accordingly
NADBusPower( )	- processes 'IRP__MJ__POWER' IRP

TABLE 6

NAD bus FDO related functions		
NADBusFDOPnp( )	- invoked when IRP__MJ__PNP is sent to NAD bus FDO - processes various minor functions according to the minor function number sent together	
	IRP__MN__START__DEVICE	-transfer NAD bus FDO to 'started' state -invokes NADBusStartFDO( )
	IRP__MN__QUERY__STOP__DEVICE	-invoked to query if NAD bus FDO can be stopped -transfer NAD bus FDO to 'stop pending' state
	IRP__MN__CANCEL__STOP__DEVICE	-invoked to cancel IRP__MN__QUERY__STOP__DEVICE
	IRP__MN__STOP__DEVICE	-stops NAD bus FDO -transfers NAD bus FDO to 'stopped' -blocks NADBusHW( ) thread
	IRP__MN__QUERY__REMOVE__DEVICE	-invoked to query if NAD bus FDO can be removed from the system
	IRP__MN__CANCEL__REMOVE__DEVICE	-invoked to cancel IRP__MN__QUERY__REMOVE__DEVICE
	IRP__MN__SURPRISE__REMOVAL	-invoked when NAD bus FDO is removed abnormally
	IRP__MN__REMOVE__DEVICE	-invoked when NAD bus FDO is removed normally
	IRP__MN__QUERY__DEVICE__RELATIONS	-passes list of NAD port PDO to PnP manager
NADBusStartFdo( )	- allocates resources to NAD bus FDO	
NADBusRemoveFdo( )	- recovers resources occupied by NAD bus FDO - removes the NADBusHW( ) thread	
NADBusGetDeviceCapabilities( )	- passes DeviceCapability data dtructure to PnP manager	

TABLE 7

NAD port PDO related functions	
NADPortPDOPnp( )	- processes minor functions related to PnP - invoked when IRP__MJ__PNP is sent to NAD port PDO Minor functions: IRP__MN__START__DEVICE IRP__MN__QUERY__STOP__DEVICE IRP__MN__CANCEL__STOP__DEVICE IRP__MN__STOP__DEVICE IRP__MN__QUERY__REMOVE__DEVICE

TABLE 7-continued

NAD port PDO related functions	
	IRP_MN_CANCEL_REMOVE_DEVICE IRP_MN_SURPRISE_REMOVAL IRP_MN_REMOVE_DEVICE IRP_MN_QUERY_CAPABILITIES IRP_MN_QUERY_ID IRP_MN_QUERY_DEVICE_RELATIONS IRP_MN_QUERY_DEVICE_TEXT IRP_MN_QUERY_RESOURCES_REQUIREMENTS IRP_MN_QUERY_RESOURCE
NADPortPDOQueryDeviceCaps( )	- returns DEVICE_CAPABILITIES data structure of NAD port
NADPortPDOQueryDeviceId( )	- returns device ID, instance ID, hardware ID of NAD port
NADPortPDOQueryDeviceText( )	- returns location and description of NAD port
NADPortPDOQueryDeviceRelations( )	- returns target device relation value
NADPortInitializePdo( )	- initialize NAD port PDO value - invoked when NAD port attached to NAD bus is detected
NADPortDestroyPdo( )	- removes NAD port PDO and recovers resources

TABLE 8

Function to detect NAD port	
NADBusHW( )	- routine for the kernel thread to detect NAD ports attached to NAD bus - periodically detects the existence of NICs - if a new NIC is detected, creates NAD port PDO and invokes NADPortInitializePdo( ) - if a NIC is detected to have been removed, removes NAD port PDO by invoking NADPortDestroyPdo( )

NAD Port Driver

A port driver is a lower-level driver that responds to a system-defined device control request or a driver-defined device I/O control request from a corresponding class driver.

The NAD port driver is capable of basic functions to initialize the driver and create an NAD port FDO and dispatch functions to process IRP passed down from the disk class driver layer. The IRP passed down from the disk class driver may contain a SCSI request block (SRB), which specifies the actual I/O command to be performed onto the SCSI device.

Tables 9 and 10 list the basic functions and some of the dispatch functions, of which roles are basically the same as those of the NAD bus driver described earlier, are presented with brief explanations.

TABLE 9

Basic functions	
DriverEntry( )	- initializes driver - registers driver functions
NADPortAddDevice( )	- invoked by PnP manager to create NAD port FDO
NADPortDriverUnload( )	- invoked when to remove driver - recovers resources

TABLE 10

Dispatch functions for initialization, creation, and removal of the NAD port	
NADPortCreateClose( )	- processes IRP_MJ_CREATE and IRP_MJ_CLOSE IRP
NADPortCleanup( )	- processes IRP_MJ_CLEANUP IRP - recovers resources
NADPortPnp( )	- processes IRP_MJ_PNP IRP
NADPortPower( )	- processes IRP_MJ_POWER IRP

In Windows 2000, a device I/O control command is included in an IRP as a device I/O control number, and the device I/O control functions are implemented in the port driver to handle the corresponding device I/O control numbers.

Besides the regular device I/O control functions in Windows 2000, additional device I/O control functions are implemented in the NAD port driver so that the NAD can be added or removed dynamically without stopping the Windows 2000 system. With conventional local disks, addition or removal of the local disk can be directly detected by the Windows 2000 system at the time of the system booting because the local disks are physically connected to the physical hardware bus. Therefore, the creation of a disk PDO for a local disk is basically initiated from the hardware interrupt at the time of the system booting. So the conventional port driver does not have to have functions that initiate addition or removal of the PDO of a disk device in the middle of the system operation.

However, in an NAD system, addition and removal of an NAD device can occur while the Windows 2000 system is running. Therefore, there should be a mechanism that can create/remove a disk PDO for the newly attached/removed disk.

The device I/O control functions implemented in the present invention handle such dynamic addition and removal of the NAD as follows. If a device control IRP that tells a new NAD hardware device is hooked up to the network is passed to the NAD port FDO, the NAD port FDO creates an NAD device PDO for the new NAD thus letting the system recognize the disk. For the removal of the NAD device, device control IRP to remove the disk is sent to and processed by NAD port FDO similarly.

The dispatch functions that handle device I/O control IRPs are summarized in the following table. Note that the device I/O control functions, NASPortFdoDeviceControl( ), NADPortPluInDevice( ), and NADPortUnpluDevice( ) are the functions particular to the present invention for the purpose of dynamic addition and removal of the NAD.

TABLE 11

NAD port device control dispatch functions	
NADPortDeviceControl()	- invoked when I/O control IRP is passed - processes IRP_MJ_DEVICE_CONTROL IRP - invokes NADPortFdoDeviceControl() for FDO control - invokes NADPortPdoDeviceControl() for PDO control

TABLE 11-continued

NADPortFdoDeviceControl()	- registers new NAD device or removes an NAD device - processes I/O control functions IOCTL_NADPORT_PLUGIN_HARDWARE Invokes NADPortPlugInDevice() to register new NAD hardware IOCTL_NADPORT_UNPLUG_HARDWARE - invokes NADPortUnplugDevice() to remove an NAD hardware
NADPortPdoDeviceControl()	- processes I/O controls for PDO - invokes I/O control functions according to the device I/O control function numbers in the IRP IOCTL_STORAGE_QUERY_PROPERTY - queries NAD device property IOCTL_GET_DISK_DRIVE_GEOMETRY - returns DISK_GEOMETRY data structure containing geometry information of the NAD device IOCTL_GET_SCSI_ADDRESS - NAD device does not use SCSI address, so sets the values of PathID and TargetID 0s and returns enumeration number of NAD device to LUN in CDB

FIG. **23**A shows the flow of IRP, SRB, and CDB where the IRP is to a SCSI disk connected to a conventional hardware bus such as PCI bus in a Windows 2000 system. A disk class driver **681** passes down to a SCSI port driver **682** and a SCSI bus driver **683** an IRP that may contain an SRB. The SRB is a data structure that contains information about the requested I/O and a command descriptor block (CDB) containing a SCSI-2 standard command. Receiving the IRP with SRB from the disk class driver, the SCSI port driver **682** and the SCSI bus driver **683** deliver the CDB extracted from the SRB to the SCSI host adapter **684** to complete an actual device I/O to a SCSI disk **685**.

In a conventional local disk, disk I/O commands are delivered to a disk controller at the host adapter using the SRB data structure. But, in the NAD system of the present invention, disk input/output commands are delivered to the NIC of the host.

FIG. **23**B shows the flow of IRP, SRB, and CDB (or some other types of I/O commands) in the NAD system. A disk class driver **691** passes down an IRP with an SRB to a NAD port driver **692** and a NAD bus driver **693**, which then deliver the CDB extracted from the SRB to NIC **694** to complete an actual device I/O to a NAD device **695** through a network **696**.

In the present invention, the NAD system supports various types of disks including SCSI and IDE. If the NAD device is composed of SCSI disks only, the CDB is delivered as is to the host NIC so that the network-attached SCSI disks can perform the requested disk I/O.

If the NAD device, however, is composed of disk devices of other type than SCSI such as IDE, the CDB must be translated into the commands that can be processed by the specific devices. The translation of the CDB, in such a case, can be done either at the NAD port driver or at the NAD device. If the translation is to be done at the NAD, the Windows 2000 host simply delivers a CDB to the host NIC as if it delivers a CDB to a SCSI disk. If the translation is to be done at the NAD port driver, the NAD port driver functions must translate the CDBs into a set of disk I/O commands appropriate to the specific hardware disk types.

The NAD system of the present invention supports both cases, and the type of the commands, i.e., CDB or hardware-specific commands, is determined at the time of the installation of the specific NAD. Some of the dispatch functions that process SRB with mandatory CDB operation codes are given in table 12 to show how the NAD port driver functions are implemented to handle the SRB and CDB in the present invention. Such SRB processing functions are required if the NAD port driver has to translate the CDB into a set of hardware specific I/O commands.

TABLE 12

NADPortInternalDeviceControl() - executes SrbFunctionExecuteScsi() when SRB_FUNCTION_EXECUTE_SCSI is passed as the SRB function value

SrbFunctionExecuteScsi()	- processes CDB - invokes CDB processing functions according to the CDB operation codes given below	
	SISCOP_TEST_UNIT_READY	- tests if an NAD device is accessible
	SCSIOP_MODE_SENSE	- returns configuration of NAD device
	SCSIOP_READ	- reads a block from NAD
	SCSIOP_WRITE	- write a block to NAD
	SCSIOP_MODE_SELECT	- sets parameter to NAD
	SCSIOP_READ_CAPACITY	- returns size of the next block or address of the last block
	SCSIOP_REASSIGN_BLOCKS	- relocates block
	SCSCIOP_RESERVE/SCSIOP_RELEASE	- changes status information
	SCSIOP_START_UNIT	- starts NAD
	SCSIOP_STOP_UNIT	- stops NAD
	SCSIOP_VERIFY	- verifies data stored in NAD

Communication between the host and the NAD

Disk I/O commands in the NAD system are delivered to the host NIC instead of the local disk host adapter because the I/O should be done over the network rather than over the bus. Windows 2000 provides a Network Driver Interface Specification (NDIS), a set of specifications defined to specify network interface drivers.

FIG. **24** shows a NDIS driver layer defined in Windows 2000. It consists of a NDIS protocol driver **701** for specifying a high-level protocol to be used, a NDIS intermediate driver **702**, an NDIS miniport **703** for managing hardware specifics, and a network interface card (NIC) **704**.

In the present invention, all the NAD port driver functions that deliver I/O commands to the NAD devices are implemented to deliver the commands to a NDIS (network driver interface specification) protocol driver layer through which the commands are delivered to the NAD devices over the network.

Upon receiving from the disk class driver the IRP containing an SRB or an I/O control command for specific disk I/O operation, the NAD port driver passes down a new IRP containing the corresponding CDB to the protocol driver. Then the protocol driver sends the CDB, which is the SCSI-2 standard I/O command, to the NAD device and, in turn, receives and handles the reply from the NAD device. Note here that if the host computer has to send some hardware specific I/O commands other than CDB as is pointed out in FIG. **23**B, the NAD port driver passes down an IRP containing the hardware specific commands instead of the CDB to the NDIS protocol driver.

The NDIS provides transport-independence for network vendors because all drivers that require communication over the network calls the NDIS interface to access the network, thus providing a ready solution for the communication between the host computer and the NAD devices in the present invention.

The actual protocol implemented in the protocol driver of the NDIS may adopt a standard protocol or a non-standard protocol. Since a standard protocol such as IP (Internet Protocol) involves an overhead, a non-standard protocol may be preferred in terms of performance and security. The present invention follows the NDIS specification of the Windows 2000 network system to implement a proprietary communication protocol into the NDIS protocol driver in order to provide a communication protocol between a Windows 2000 host and NAD devices to reliably handle the NAD I/O commands.

NAD Device

The technical constitution of the NAD device running under the Windows family of operating systems is the same as that of the NAD device running under the UNIX family of operating systems shown in FIG. **14**.

Advantages of the NAD System over NAS and SAN

Either running the UNIX or Windows family of operating systems, the NAD system of the present invention has numerous advantages over the NAS system and the SAN system. Unlike the NAS system that provides file storage service by way of an additional file server, the NAD device is attached to a host computer as if it is a local disk connected to the system bus of the host. Unlike the SAN system, the NAD device of the present invention is simply plugged into a network port without requiring any additional special switch or network equipment. Therefore, the NAD system provides better user convenience, system flexibility, scalability, economy, and performance.

All the disk-related operations, including formatting, partitioning, sharing, and mounting, can be done to NAD devices just as they can be done to a local disk. Since NAD devices are directly available to the host as local disks, the NAD system provides better manageability and user convenience. In the NAS system, addition, deletion, or any change to the disk configuration should be consulted to the NAS operating system through human or software intervention. In the NAD system, addition or deletion of an NAD device is instantly achieved by plugging or unplugging the NAD device to and from a network port. The NAD system even provides a superior user convenience in installing and uninstalling the disks, eliminating the need of opening and closing the case of the host computer.

The NAD system provides almost unlimited scalability to the disk capacity. The number of NAD devices that can be attached to the network is virtually unlimited, whereas the number of disks available through the NAS system is severely limited because of an economical reason and the inconvenience involved in the management of the multitudes of NAS servers.

The NAD system is intrinsically more economical than the NAS or SAN system because each NAD device does not employ file server software and other additional special hardware equipment.

Media Changeable NAD system

An NAD system of the present invention can be alternatively implemented as a media changeable storage device. A media changeable storage device is a special storage device that is physically separated two parts, one being the media containing the data and the other being the driver performing an I/O operation to the media. Floppy disk drivers, CD-ROM drivers are examples of media changeable storage devices. Whether a media is installed or not, a media changeable storage device can be registered to a host computer so that a media such as a diskette can be inserted into a driver dynamically.

Since NAD devices can be plugged in or removed from a network port dynamically, a virtual driver that uses NAD as a media can be implemented in the form of a media changeable storage device. Windows 2000 provides the changer class driver model to implement a media changeable storage device. In order to implement a media changeable NAD system, a class driver for the NAD system is implemented according to the model of the changer class driver of Windows 2000. The two lower-level drivers, i.e. the NAD port driver and the NAD bus driver, are used to implement such media changeable NAD system.

Alternative Embodiment using Converter and Counter-Converter

Instead of using a network interface card (NIC) and new virtual host bus adapter, the network attached disk of the present invention may be implemented by providing in the host side a protocol converter that converts storage commands into data link frames containing the storage commands so that the frames can be sent through a network, and by providing in the device side a counter-converter that converts the data link frames containing the storage commands received through the network into the storage commands.

Since a converter is a specialized network interface, the converter encapsulates the I/O commands and data to data link frame so as transmit them to an I/O device through a network without the overhead of processing communication protocols in general.

Tape System, CD Juke Box

The kinds of storage devices that can be directly connected to a network using the interface of the present system are not limited to disk systems. Tape systems and CD drives use IDE or SCSI interface, the same bus interface as disk systems. For example, the present invention may be used to connect multiple CD drives directly to a network, enabling a cost-effective implementation of a CD-Juke box.

While the invention has been described with reference to preferred embodiments, it is not intended to be limited to those embodiments. It will be appreciated by those of ordinary skilled in the art that many modifications can be made to the structure and form of the described embodiments without departing from the spirit and scope of this invention.

What is claimed is:

1. A network-attached disk (NAD) device configured to be connected through a general purpose front-end network to a host having a system bus, wherein the host has a virtual host bus adapter that recognizes the device as a local device connected directly to the system bus of the host, the NAD device comprising:

a network adapter for receiving a disk access command in data link frames through the general purpose front-end network;

a disk controller, connected to the network adapter, for executing the disk access command;

a disk for storing data; and

a controller, connected to the network adapter and the disk controller, for controlling the operation of the NAD device;

wherein access to the disk is operatively controlled by the disk controller, no disk access command is required to be routed through a server, the NAD device is configured to be automatically discovered by the host when the NAD device is connected to the general purpose front-end network, the NAD device is not directly connected to a system bus of the host, and the NAD device is configured to be recognized by the virtual host bus driver created by a NAD device driver of the host, the NAD device driver comprising a device file and device driver routines for the device driver to register the device driver to the host.

2. The NAD device of claim **1**, wherein the network runs Ethernet.

3. The NAD device of claim **1**, wherein said disk is formatted as a local disk.

4. The NAD device of claim **1**, wherein said disk is partitioned as a local disk.

5. The NAD device of claim **1**, wherein the network adapter has a physical network interface for receiving data from a host and a media access control (MAC) controller.

6. The NAD device of claim **1**, wherein the controller has a state machine for controlling the operation of the NAD device.

7. The NAD device of claim **1**, wherein the controller has a filter for controlling access to the disk.

8. The NAD device of claim **1**, wherein the disk is partitioned into a plurality of disk partitions.

9. The NAD device of claim **8**, wherein each disk partition is controlled by a separate driver.

10. The NAD device of claim **1**, wherein the NAD device is configured for connections with the NAD device driver having a network connection setting thread for making a connection between the device driver and the NAD device.

11. The NAD device of claim **1**, wherein the NAD device is configured to be controlled by the NAD device driver further including: a bus driver for creating the virtual host adapter to access the NAD device as a local device connected directly to the system bus of the host; and a port driver for communicating the disk access command from the host to the NAD device through a network port.

12. A network-attached disk (NAD) device configured to be connected through a general purpose front-end network to a host computer having a system bus, the NAD device comprising:

a network adapter for receiving a disk access command through the general purpose front-end network, said network adapter including:

a physical network interface for interfacing with the general purpose front-end network to receive a disk I/O request packet from the host computer, and

a media access control (MAC) controller connected to the physical network interface to extract necessary data from the disk I/O request packet;

a disk controller, connected to the network adapter, for executing the disk access command;

a disk for storing data; and

a controller, connected to the network adapter and the disk controller, for controlling the operation of the NAD device;

wherein no disk access command is required to be routed through a server, the NAD device is not directly connected to a system bus of the host, and the NAD device is configured to be automatically recognized by a virtual host bus driver created by a NAD device driver of the host, the NAD device driver comprising a device file and device driver routines for the device driver to register the device driver to the host.

13. The NAD device of claim **12**, wherein automatic discovery of the NAD device occurs when it is connected to the general purpose front-end network.

14. The NAD device of claim **12**, wherein the general purpose front-end network runs Ethernet.

15. The NAD device of claim **12**, wherein said disk is formatted as a local disk.

16. The NAD device of claim **12**, wherein said disk is partitioned as a local disk.

17. The NAD device of claim **12**, wherein the controller has a state machine for controlling the operation of the NAD device.

18. The NAD device of claim **12**, wherein the controller has a filter for controlling access to the disk.

19. The NAD device of claim **12**, wherein the disk is partitioned into a plurality of disk partitions.

20. The NAD device of claim **19**, wherein each disk partition is controlled by a separate driver.

21. The NAD device of claim **1**, wherein the NAD device is configured to be accessed by the NAD device driver having a device accessing thread for accessing the NAD device.

22. The NAD device of claim **1**, wherein the NAD device is configured to be identified by the NAD device driver having a device searching thread for identifying a device attached to the network.

* * * * *

EXHIBIT J

US007849153B2

(12) **United States Patent**
Kim

(10) **Patent No.:** **US 7,849,153 B2**
(45) **Date of Patent:** **Dec. 7, 2010**

(54) **DISK SYSTEM ADAPTED TO BE DIRECTLY ATTACHED**

(75) Inventor: **Han-Gyoo Kim**, Irvine, CA (US)

(73) Assignee: **Zhe Khi Pak**, Moscow (RU)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 1574 days.

(21) Appl. No.: **11/153,985**

(22) Filed: **Jun. 16, 2005**

(65) **Prior Publication Data**

US 2006/0010287 A1 Jan. 12, 2006

Related U.S. Application Data

(62) Division of application No. 09/974,082, filed on Oct. 9, 2001, now Pat. No. 7,792,923.

(60) Provisional application No. 60/240,344, filed on Oct. 13, 2000.

(51) **Int. Cl.**
G06F 15/16 (2006.01)
G06F 3/00 (2006.01)
G06F 12/00 (2006.01)

(52) **U.S. Cl.** **709/217**; 709/218; 709/236; 709/246; 709/250; 710/5; 710/36; 711/110

(58) **Field of Classification Search** 709/217, 709/218, 236, 246, 250; 711/110; 710/5
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,329,619 A 7/1994 Page et al.
5,426,427 A 6/1995 Chinnock et al.
5,459,857 A * 10/1995 Ludlam et al. 714/6
5,463,772 A 10/1995 Thompson et al.
5,513,314 A 4/1996 Kandasamy et al.
5,524,247 A 6/1996 Mizuno

(Continued)

FOREIGN PATENT DOCUMENTS

DE 19610840 9/1997

(Continued)

OTHER PUBLICATIONS

Blunden et al., "Storage Network Virtualization: What's it all about?", ibm.com/redbooks, XP-002286341, pp. 1-110, Dec. 2000.

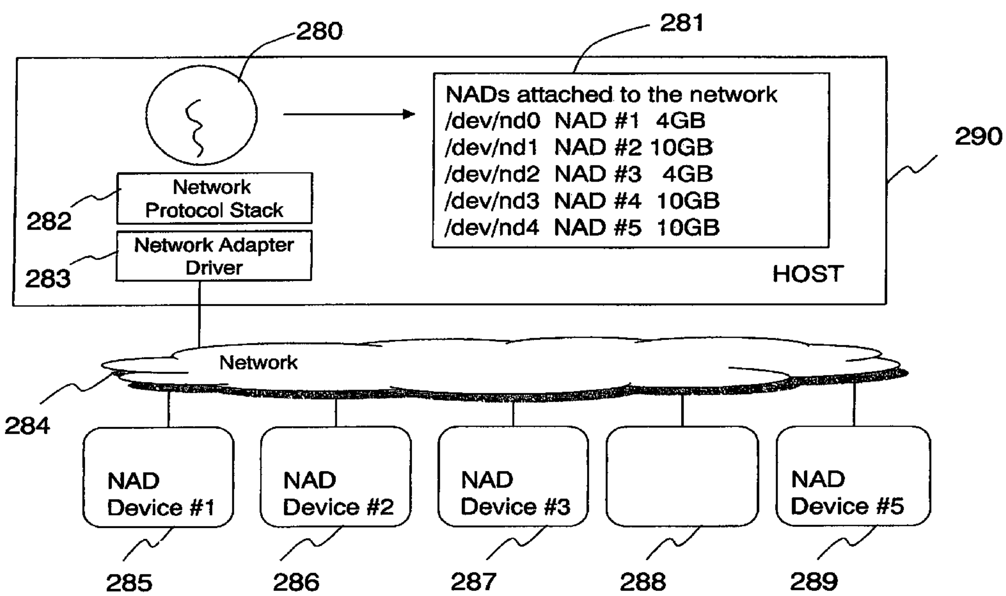
(Continued)

Primary Examiner—Saleh Najjar
Assistant Examiner—Vitali Korobov
(74) *Attorney, Agent, or Firm*—Rothwell, Figg, Ernst & Manbeck, P.C.

(57) **ABSTRACT**

A network-attached disk (NAD) system is disclosed that includes an NAD device for receiving a disk access command from a host through a network, and a device driver at the host for controlling the NAD device through the network, where the device driver creates a virtual host bus adapter so that the host recognizes the NAD device as if it is a local device to the host. The host may run the UNIX or Windows family of operating systems. The NAD device includes a disk for storing data, a disk controller for controlling the disk, and a network adapter for receiving a disk access command from the host through a network port.

11 Claims, 24 Drawing Sheets



U.S. PATENT DOCUMENTS

5,566,331	A *	10/1996	Irwin et al.	1/1
5,721,818	A	2/1998	Hanif et al.	
5,774,660	A	6/1998	Brendel et al.	
5,781,550	A	7/1998	Templin et al.	
5,812,930	A	9/1998	Zavrel	
5,838,916	A	11/1998	Domenikos et al.	
5,889,942	A	3/1999	Orenshteyn	
5,941,972	A *	8/1999	Hoese et al.	710/315
5,987,523	A	11/1999	Hind et al.	
5,987,627	A	11/1999	Raqlings, III	
5,999,808	A	12/1999	Ladue	
6,047,307	A	4/2000	Radko	
6,085,234	A	7/2000	Pitts et al.	
6,128,644	A	10/2000	Nozaki	
6,128,690	A	10/2000	Purcell et al.	
6,167,490	A	12/2000	Levy et al.	
6,175,869	B1	1/2001	Ahuja et al.	
6,216,202	B1 *	4/2001	D'Errico	711/112
6,314,465	B1	11/2001	Paul et al.	
6,317,775	B1	11/2001	Colie et al.	
6,327,594	B1	12/2001	Van Huben et al.	
6,345,300	B1	2/2002	Bakashi et al.	
6,347,095	B1	2/2002	Tang et al.	
6,351,776	B1 *	2/2002	O'Brien et al.	709/245
6,356,915	B1 *	3/2002	Chtchetkine et al.	707/823
6,360,265	B1	3/2002	Falck et al.	
6,366,988	B1 *	4/2002	Skiba et al.	711/165
6,389,432	B1 *	5/2002	Pothapragada et al.	1/1
6,393,569	B1	5/2002	Orenshteyn	
6,404,766	B1	6/2002	Kitai et al.	
6,421,753	B1 *	7/2002	Hoese et al.	710/315
6,449,647	B1	9/2002	Colby et al.	
6,470,389	B1	10/2002	Chung et al.	
6,510,164	B1	1/2003	Ramaswamy et al.	
6,518,965	B2	2/2003	Dye et al.	
6,523,066	B1	2/2003	Montroy et al.	
6,529,996	B1 *	3/2003	Nguyen et al.	711/114
6,539,446	B1	3/2003	Chan	
6,578,111	B1	6/2003	Damron et al.	
6,594,677	B2	7/2003	Davis et al.	
6,598,068	B1	7/2003	Clark	
6,609,167	B1	8/2003	Bastiani et al.	
6,647,016	B1	11/2003	Isoda et al.	
6,732,104	B1	5/2004	Weber	
6,760,783	B1	7/2004	Berry	
6,807,581	B1 *	10/2004	Starr et al.	709/250
6,823,458	B1	11/2004	Lee et al.	
6,834,326	B1 *	12/2004	Wang et al.	711/114
6,894,981	B1	5/2005	Coile et al.	
6,941,576	B2	9/2005	Amit	
6,985,927	B2 *	1/2006	O'Brien et al.	709/213
7,010,303	B2	3/2006	Lewis et al.	
7,069,312	B2	6/2006	Kostic et al.	
7,069,350	B2	6/2006	Fujita et al.	
7,076,690	B1	7/2006	Todd et al.	
7,124,128	B2	10/2006	Springer et al.	
7,251,704	B2	7/2007	Solomon et al.	
7,254,578	B2	8/2007	Devarakonda et al.	
7,277,955	B2	10/2007	Elliott	
7,376,133	B2	5/2008	Gettala et al.	
7,383,229	B2	6/2008	Jacoby	
2001/0044879	A1 *	11/2001	Moulton et al.	711/114
2002/0069245	A1	6/2002	Kim	
2002/0103889	A1 *	8/2002	Markson et al.	709/223
2003/0014569	A1	1/2003	Kim	
2003/0018403	A1	1/2003	Braun et al.	
2003/0028614	A1	2/2003	Jeon	
2003/0172149	A1	9/2003	Edsall et al.	
2003/0225834	A1	12/2003	Lee et al.	
2004/0068563	A1	4/2004	Ahuja et al.	
2004/0117813	A1	6/2004	Karaoguz et al.	
2004/0220933	A1	11/2004	Walker	
2005/0042591	A1	2/2005	Bloom et al.	
2005/0110768	A1	5/2005	Marriott et al.	
2005/0149682	A1	7/2005	Kim	
2005/0193017	A1	9/2005	Kim	
2005/0193189	A1	9/2005	Kim	
2006/0004935	A1	1/2006	Seto et al.	
2006/0045130	A1	3/2006	Kim	
2006/0067356	A1	3/2006	Kim	
2006/0069884	A1	3/2006	Kim	
2006/0155805	A1	7/2006	Kim	
2007/0008988	A1	1/2007	Kim et al.	

FOREIGN PATENT DOCUMENTS

DE	10038142	2/2001
JP	10-113469	5/1998
JP	10-271562	10/1998
JP	11007404	1/1999
JP	11-114224	4/1999
KR	10-2000-72493	12/2000
KR	10-2001-0088528	9/2001
WO	WO99/03297	7/1999
WO	WO00/29529	5/2000

OTHER PUBLICATIONS

Klein, Yaron, "Storage Virtualization with iSCSI Protocol", Internet Draft, XP-015030964, pp. 1-15, Nov. 2, 2000.

Schulz, Greg, "SAN and NAS; Complementary Technologies", http://www.mti.com/white_papers/WP20002.pdf, XP-002201566, pp. 1-7, May 1, 2000.

Supplementary European Search Report, Application No. 01272932. 3, 4 pages, Nov. 5, 2007.

Japanese Office Action, App. No. 555298/2002, Jan. 9, 2007.

Japanese Office Action, App. No. 513300/2003, Feb. 6, 2007.

* cited by examiner

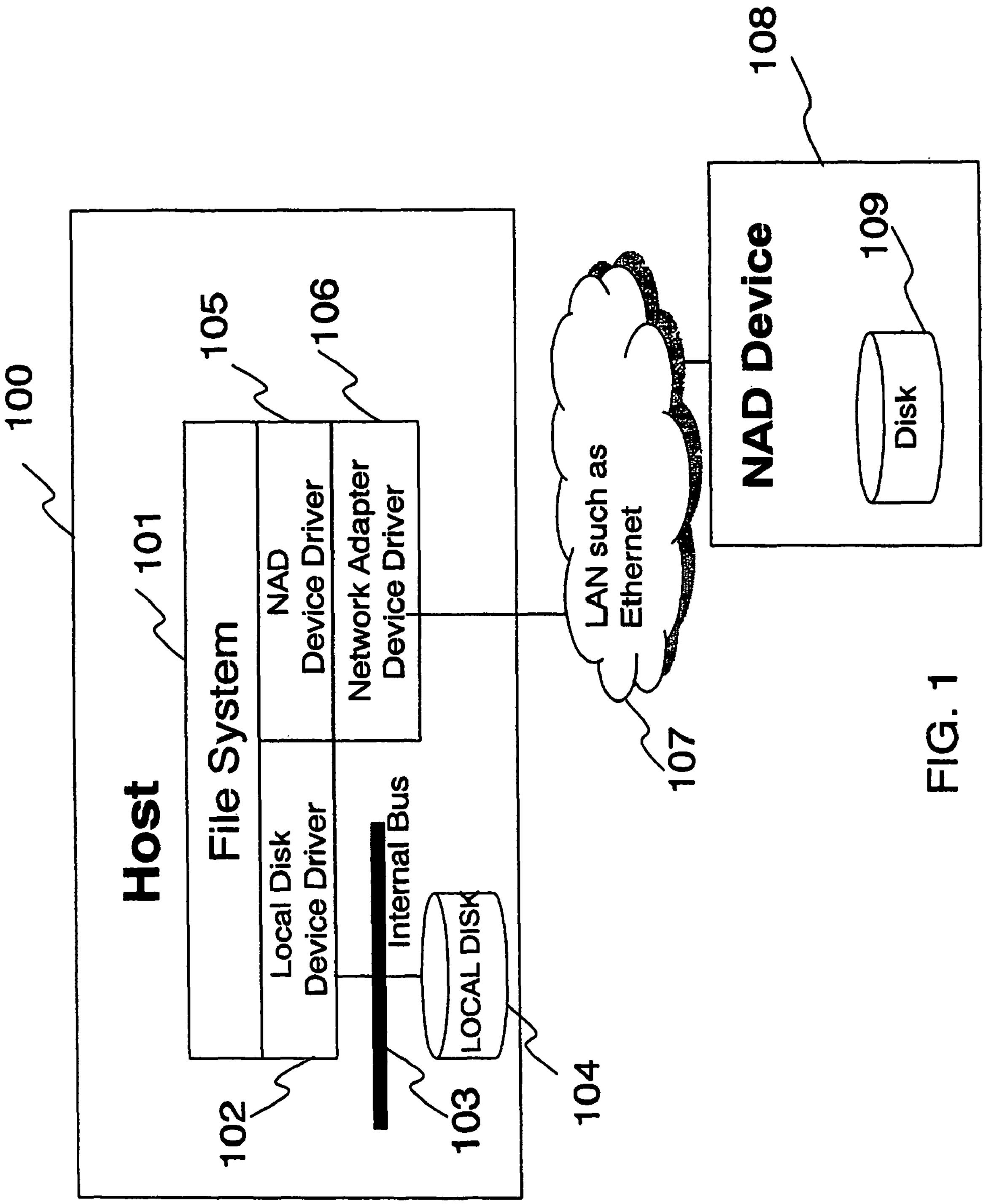
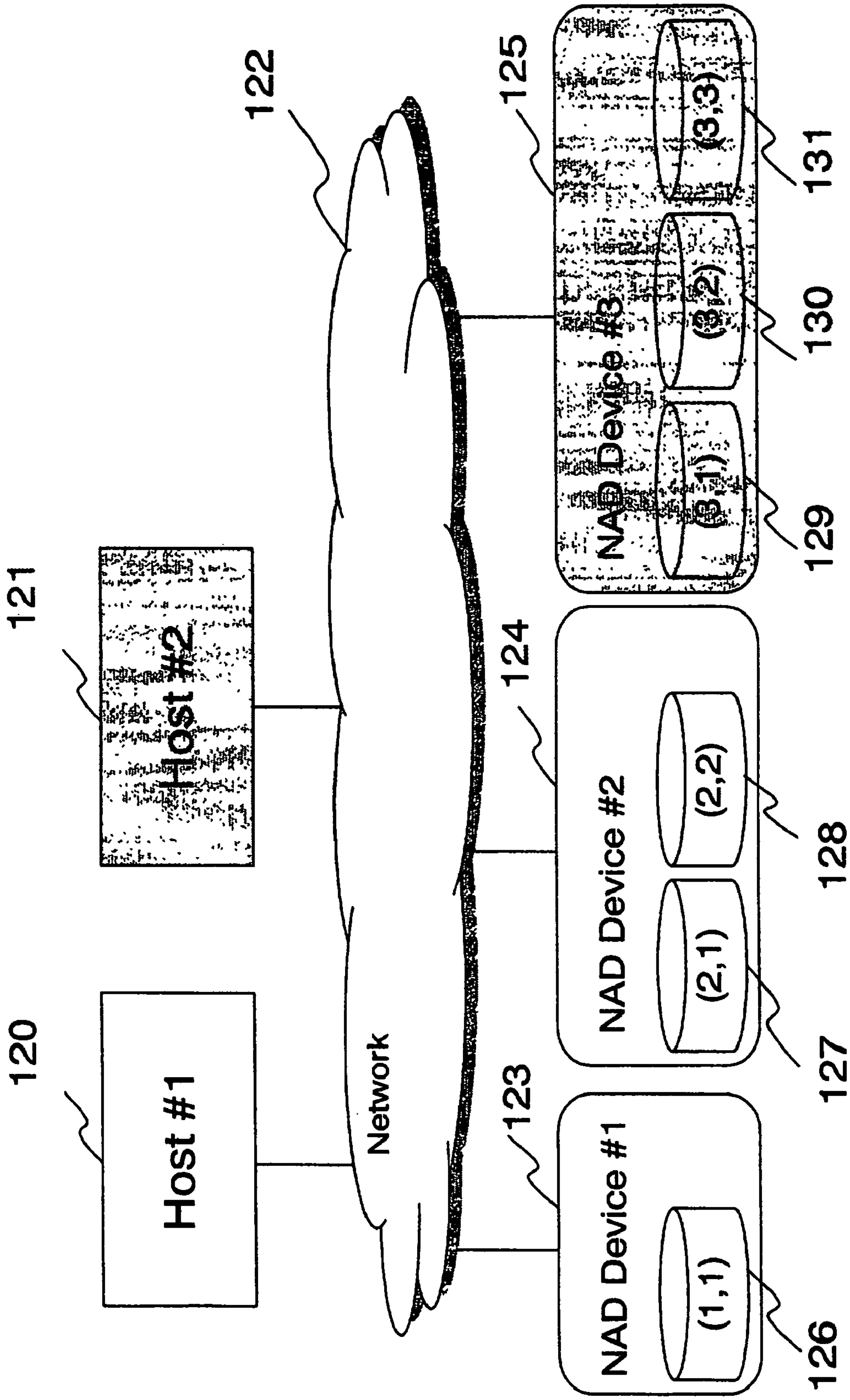

100
Host
101
File System
102
Local Disk Device Driver
105
NAD Device Driver
106
Network Adapter Device Driver
Internal Bus
103
LOCAL DISK
104
LAN such as Ethernet
107
108
NAD Device
109
Disk

FIG. 1



120
Host #1
121
Host #2
122
Network
123
NAD Device #1
(1,1)
126
124
NAD Device #2
(2,1)
(2,2)
127
128
125
NAD Device #3
(3,1)
(3,2)
(3,3)
129
130
131

FIG. 2

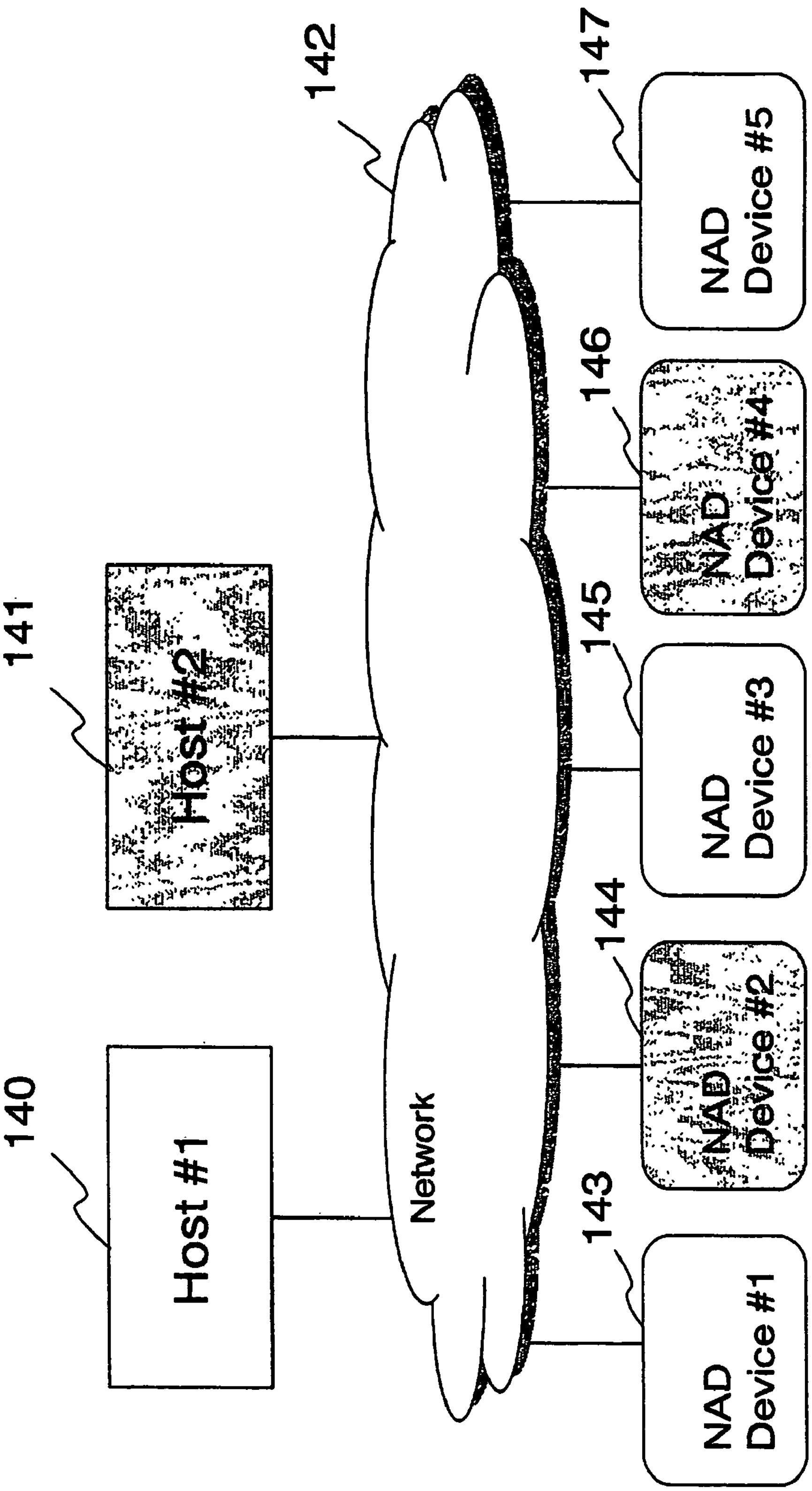

140
Host #1
141
Host #2
142
Network
143
NAD Device #1
144
NAD Device #2
145
NAD Device #3
146
NAD Device #4
147
NAD Device #5

FIG. 3

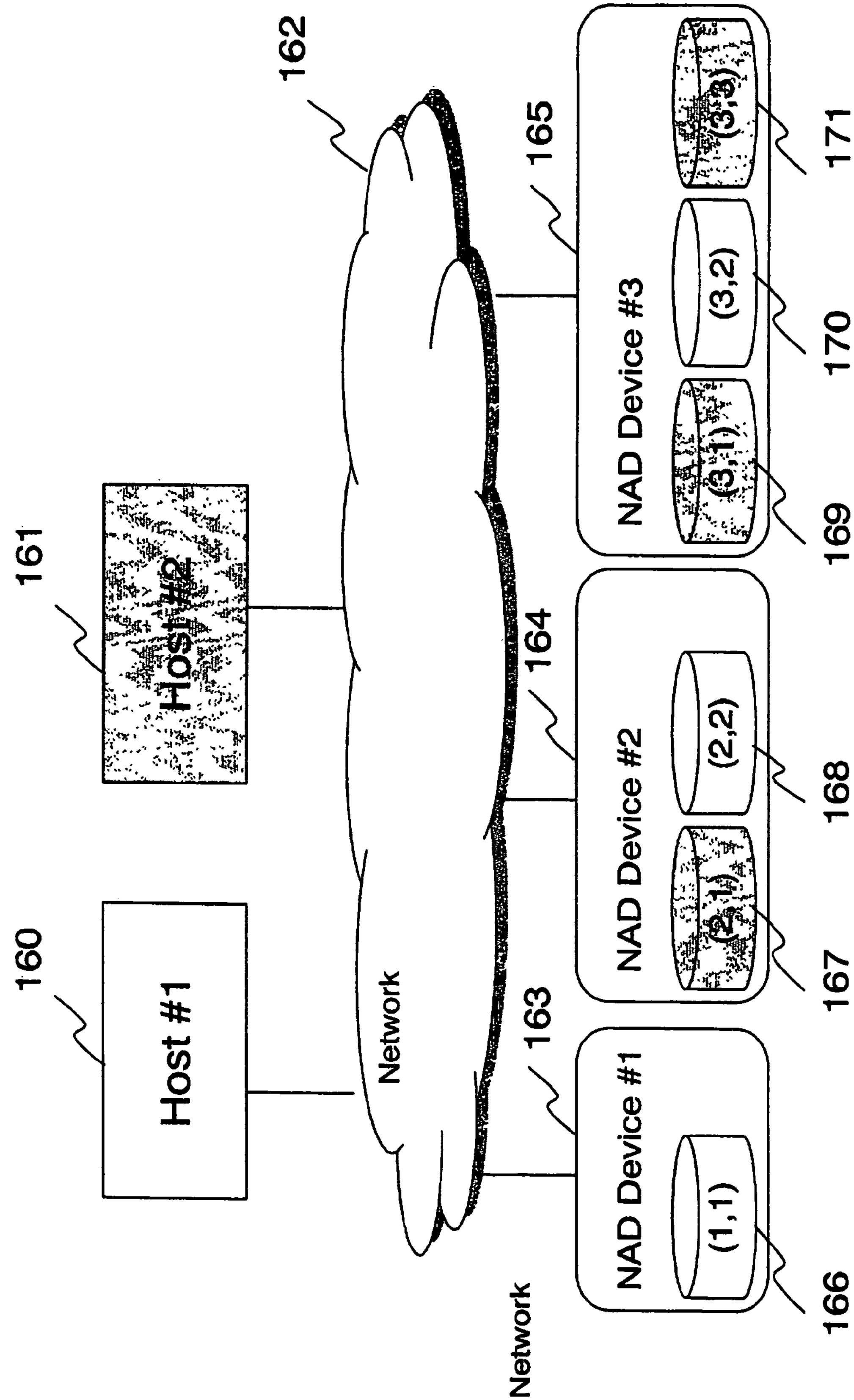

160
161
162
Host #1
Host #2
Network
163
164
165
NAD Device #1
NAD Device #2
NAD Device #3
(1,1)
(2,1)
(2,2)
(3,1)
(3,2)
(3,3)
166
167
168
169
170
171
Network

FIG. 4

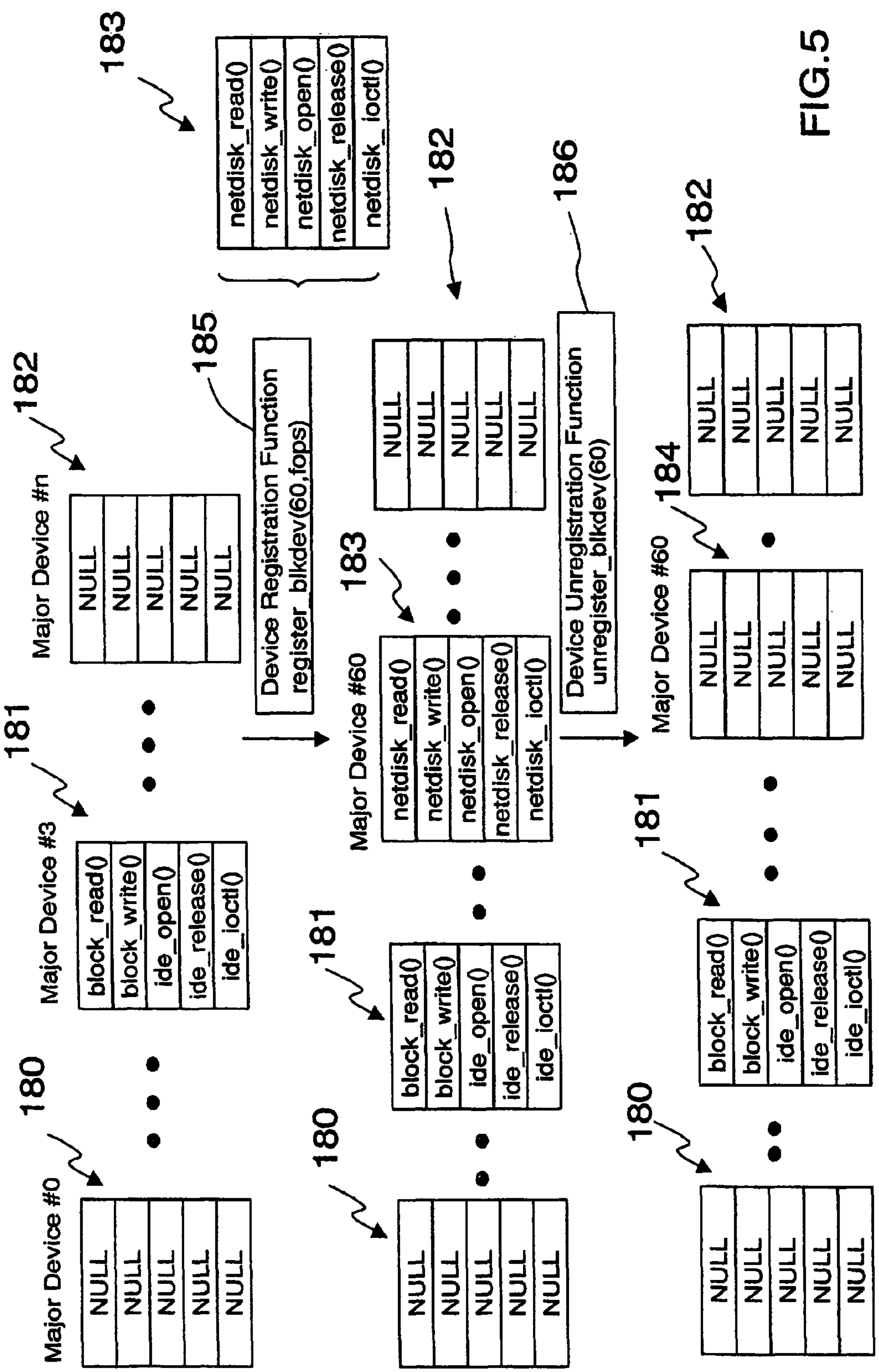

Major Device #0
NULL
NULL
NULL
NULL
NULL
180
Major Device #3
block_read()
block_write()
ide_open()
ide_release()
ide_ioctl()
181
Major Device #n
NULL
NULL
NULL
NULL
NULL
182
Device Registration Function
register_blkdev(60,fops)
185
netdisk_read()
netdisk_write()
netdisk_open()
netdisk_release()
netdisk_ioctl()
183
Major Device #60
netdisk_read()
netdisk_write()
netdisk_open()
netdisk_release()
netdisk_ioctl()
183
182
181
180
Device Unregistration Function
unregister_blkdev(60)
186
Major Device #60
NULL
NULL
NULL
NULL
NULL
184
182
181
180
block_read()
block_write()
ide_open()
ide_release()
ide_ioctl()

FIG.5

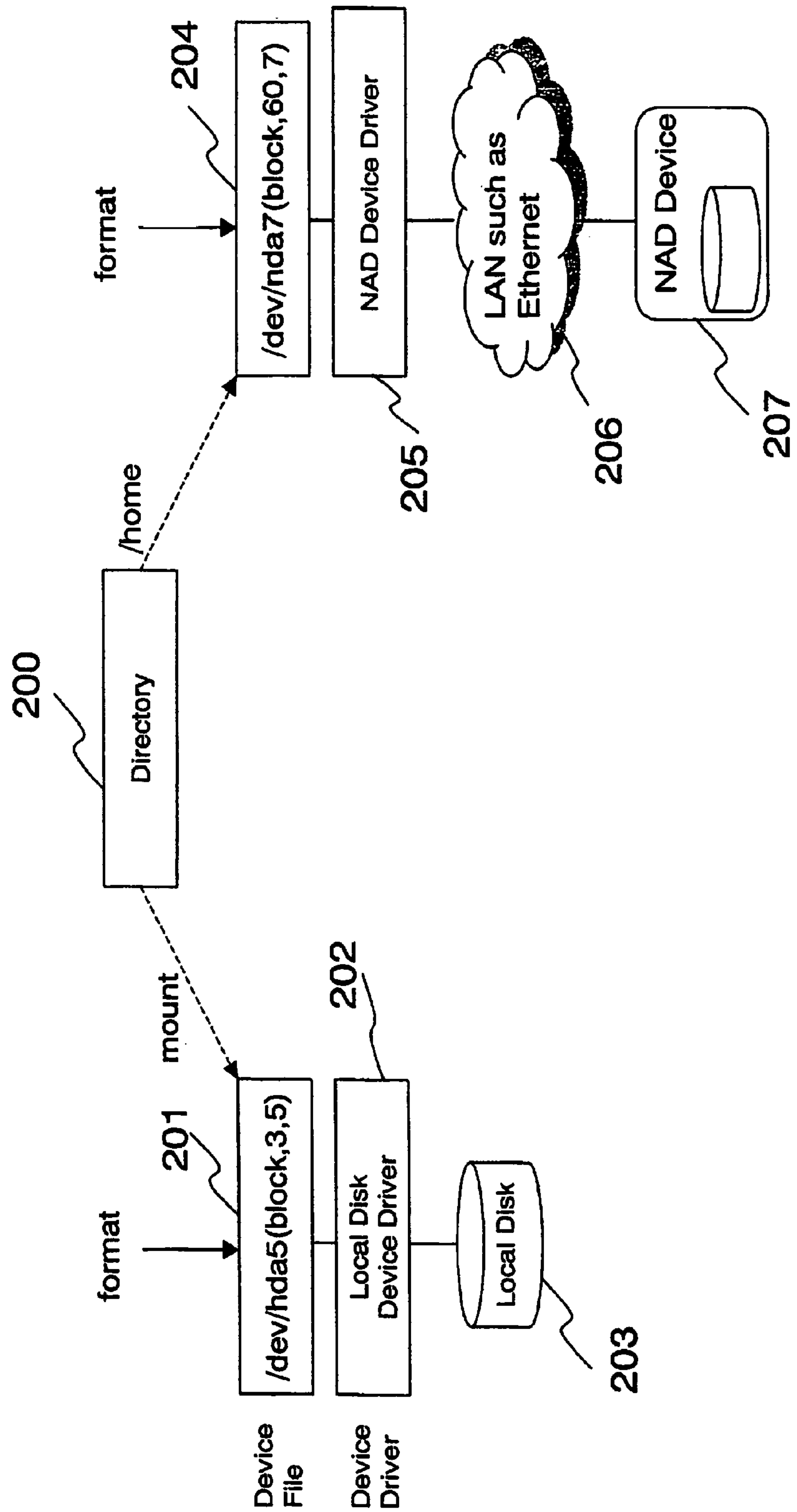

Directory
200
/home
mount
format
/dev/nda7(block,60,7)
204
NAD Device Driver
205
LAN such as Ethernet
206
NAD Device
207
format
/dev/hda5(block,3,5)
201
Local Disk Device Driver
202
Local Disk
203
Device File
Device Driver

FIG. 6

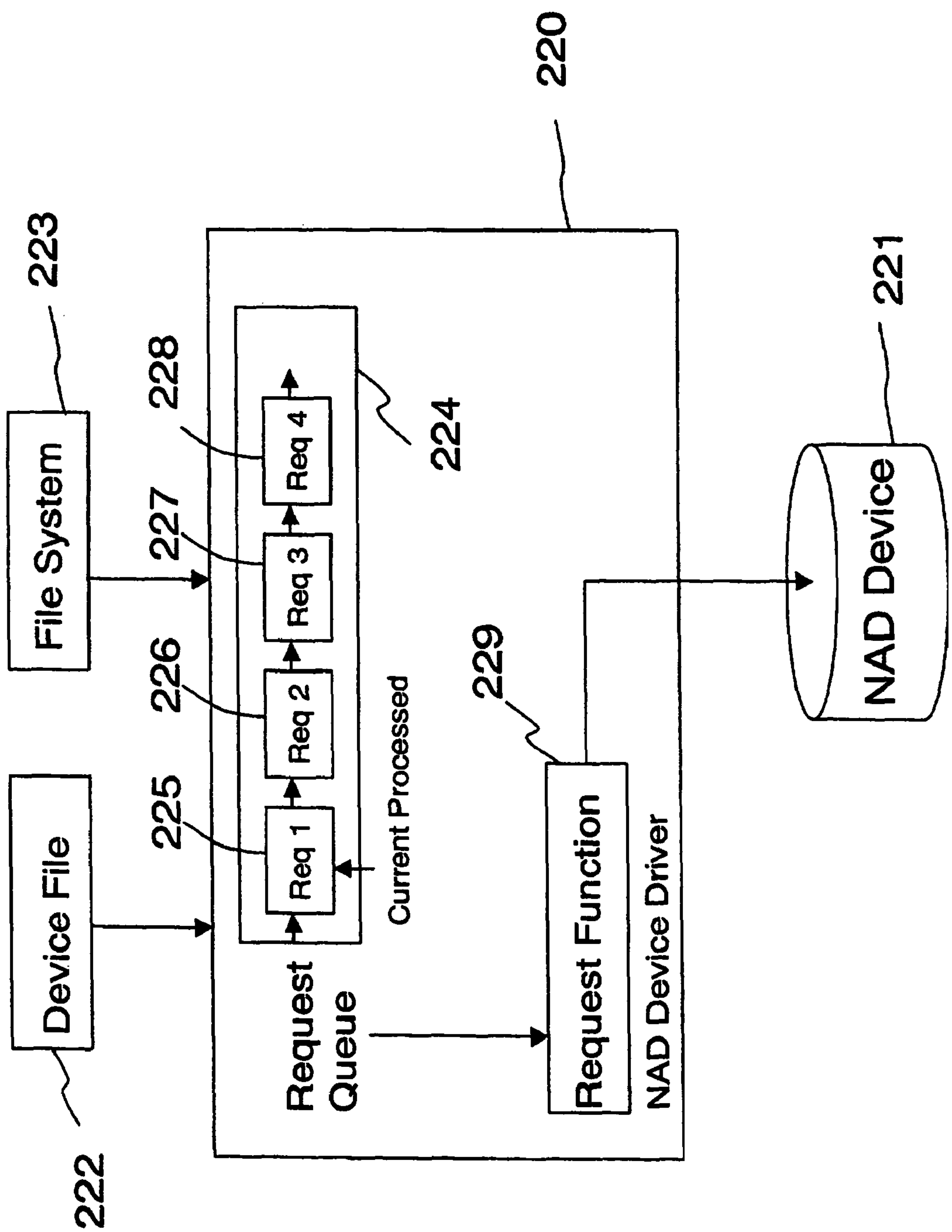

220
223
File System
222
Device File
224
Request
Queue
Req 1
Req 2
Req 3
Req 4
225
226
227
228
Current Processed
229
Request Function
NAD Device Driver
221
NAD Device

FIG. 7

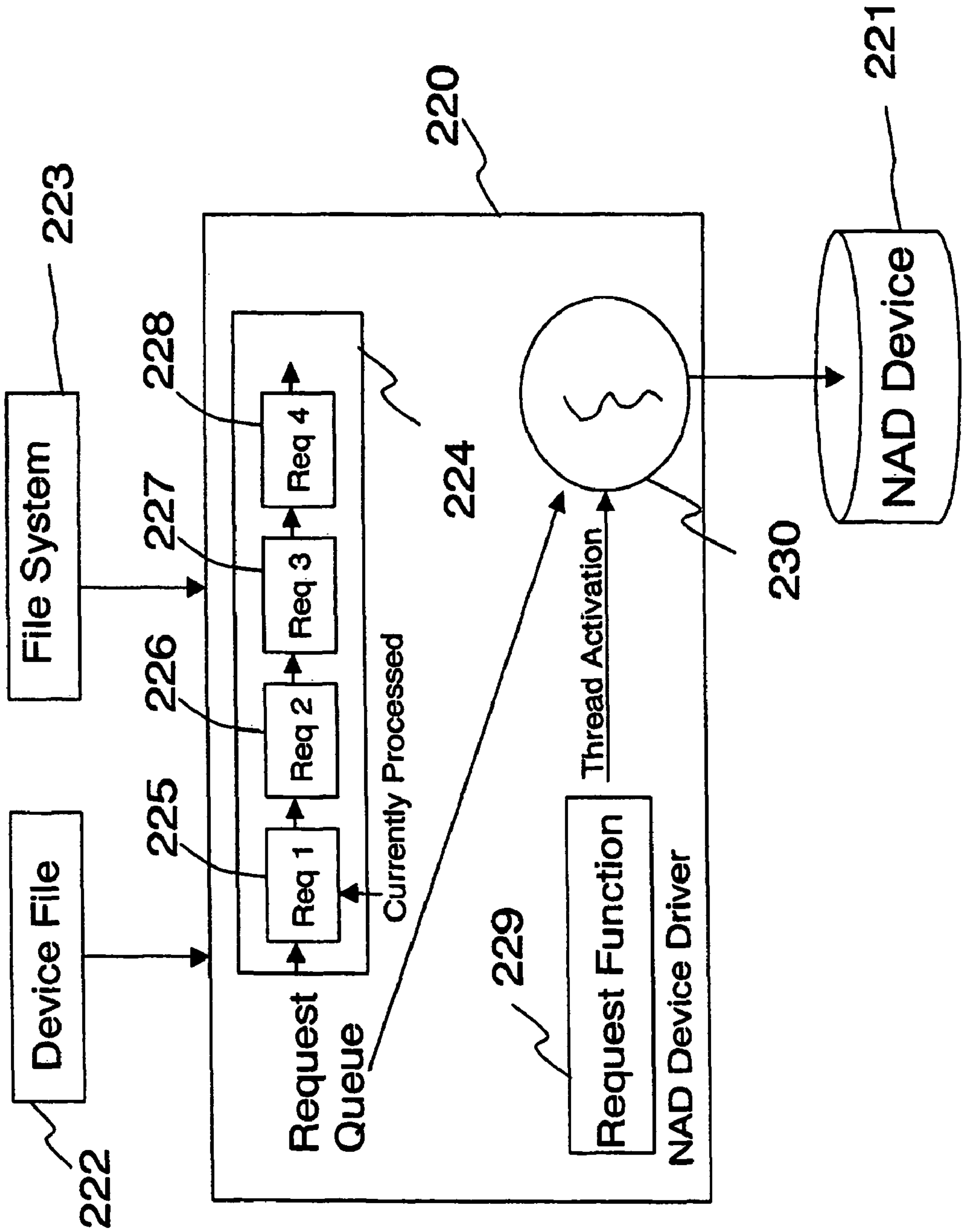

Device File
File System
Request Queue
Req 1
Req 2
Req 3
Req 4
Currently Processed
Thread Activation
Request Function
NAD Device Driver
NAD Device
220
221
222
223
224
225
226
227
228
229
230

FIG. 8

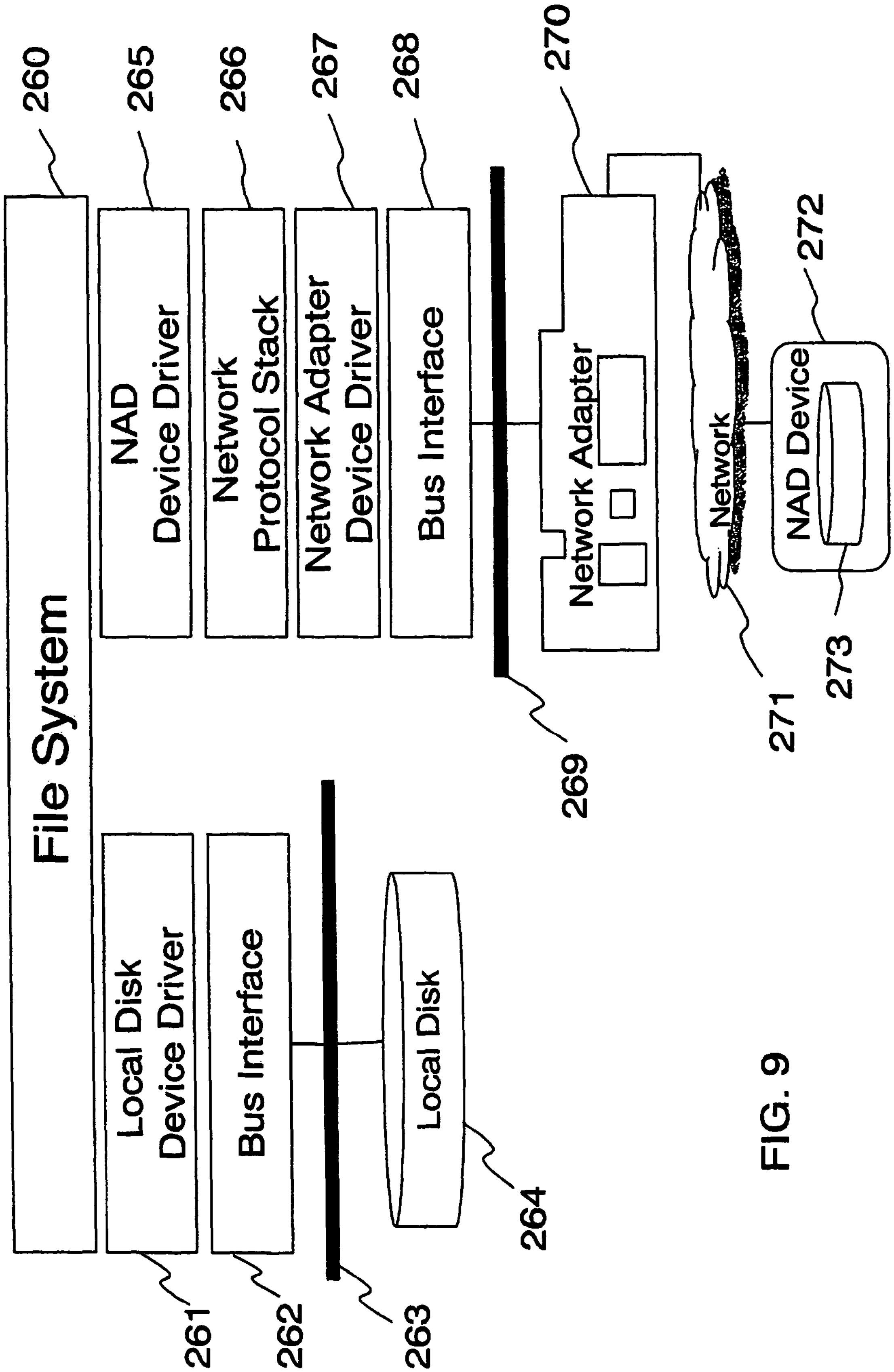

File System
NAD Device Driver
Network Protocol Stack
Network Adapter Device Driver
Bus Interface
Network Adapter
Network
NAD Device
Local Disk Device Driver
Bus Interface
Local Disk
260
265
266
267
268
270
272
269
271
273
261
262
263
264

FIG. 9

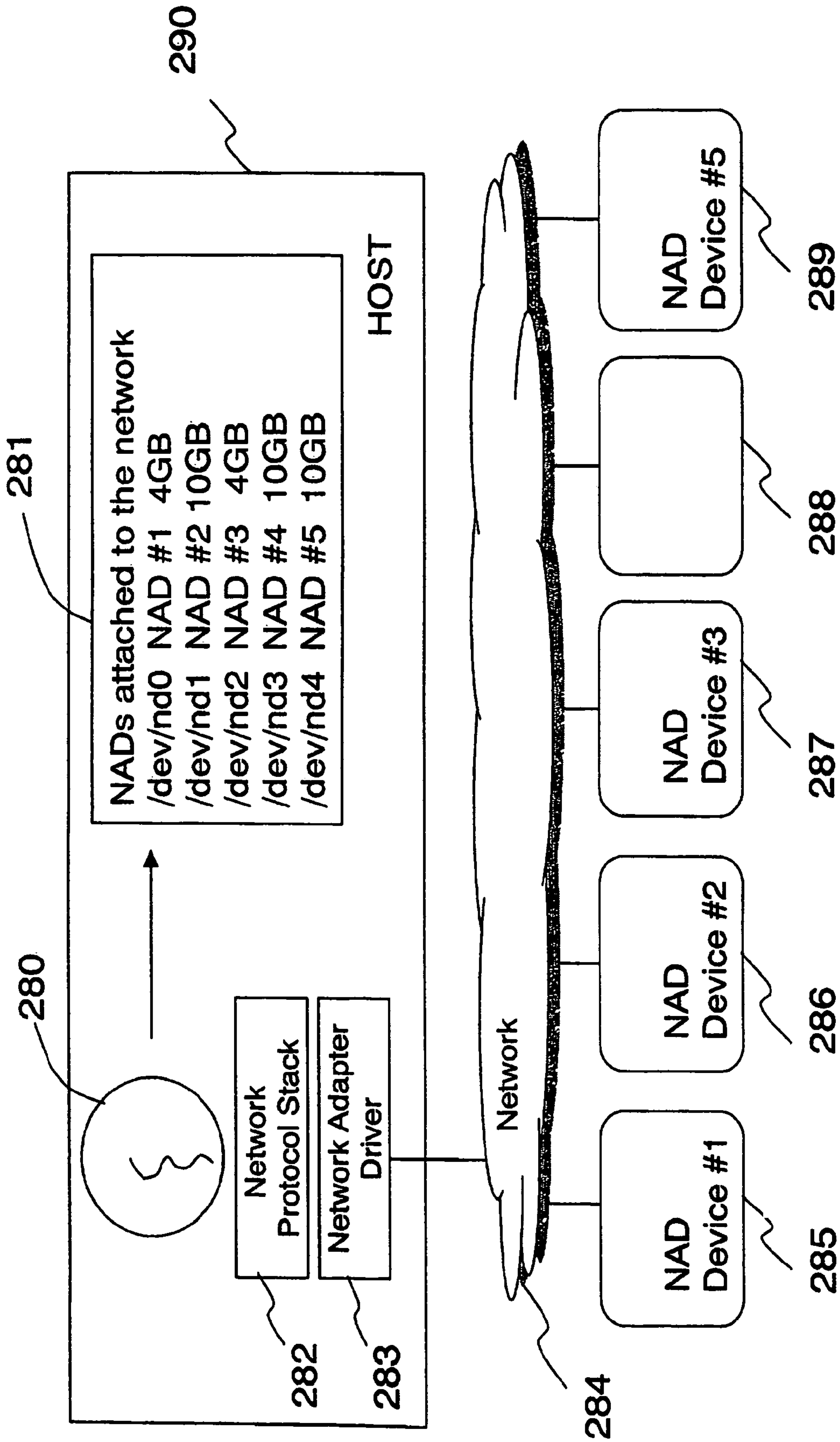

290
281
NADs attached to the network
/dev/nd0 NAD #1 4GB
/dev/nd1 NAD #2 10GB
/dev/nd2 NAD #3 4GB
/dev/nd3 NAD #4 10GB
/dev/nd4 NAD #5 10GB
HOST
280
Network Protocol Stack
Network Adapter Driver
282
283
Network
284
NAD Device #1
NAD Device #2
NAD Device #3
NAD Device #5
285
286
287
288
289

FIG. 10

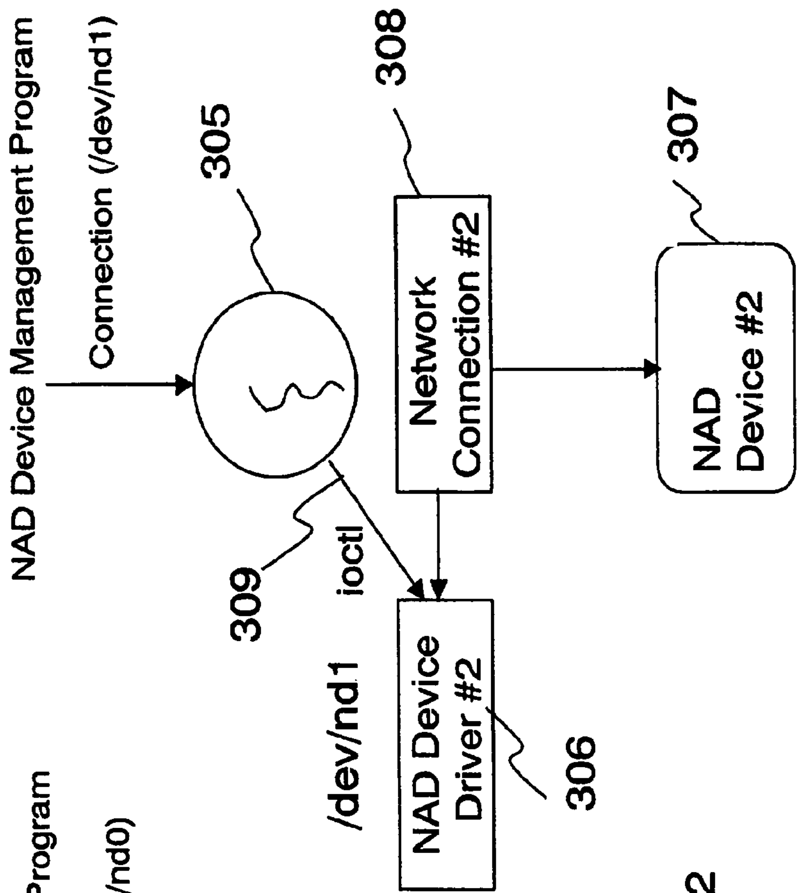


FIG. 11B

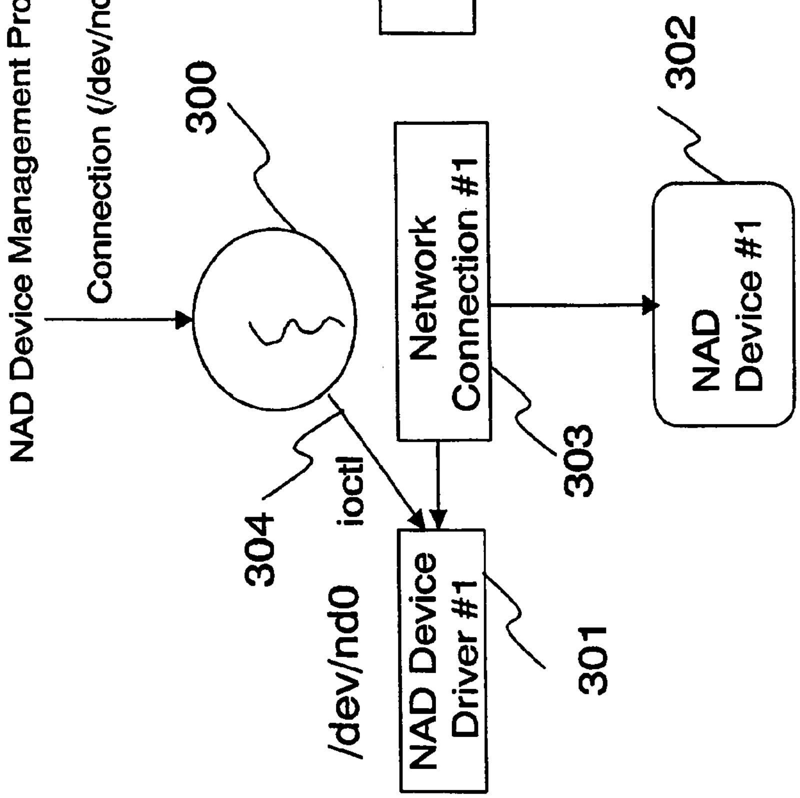


FIG. 11A

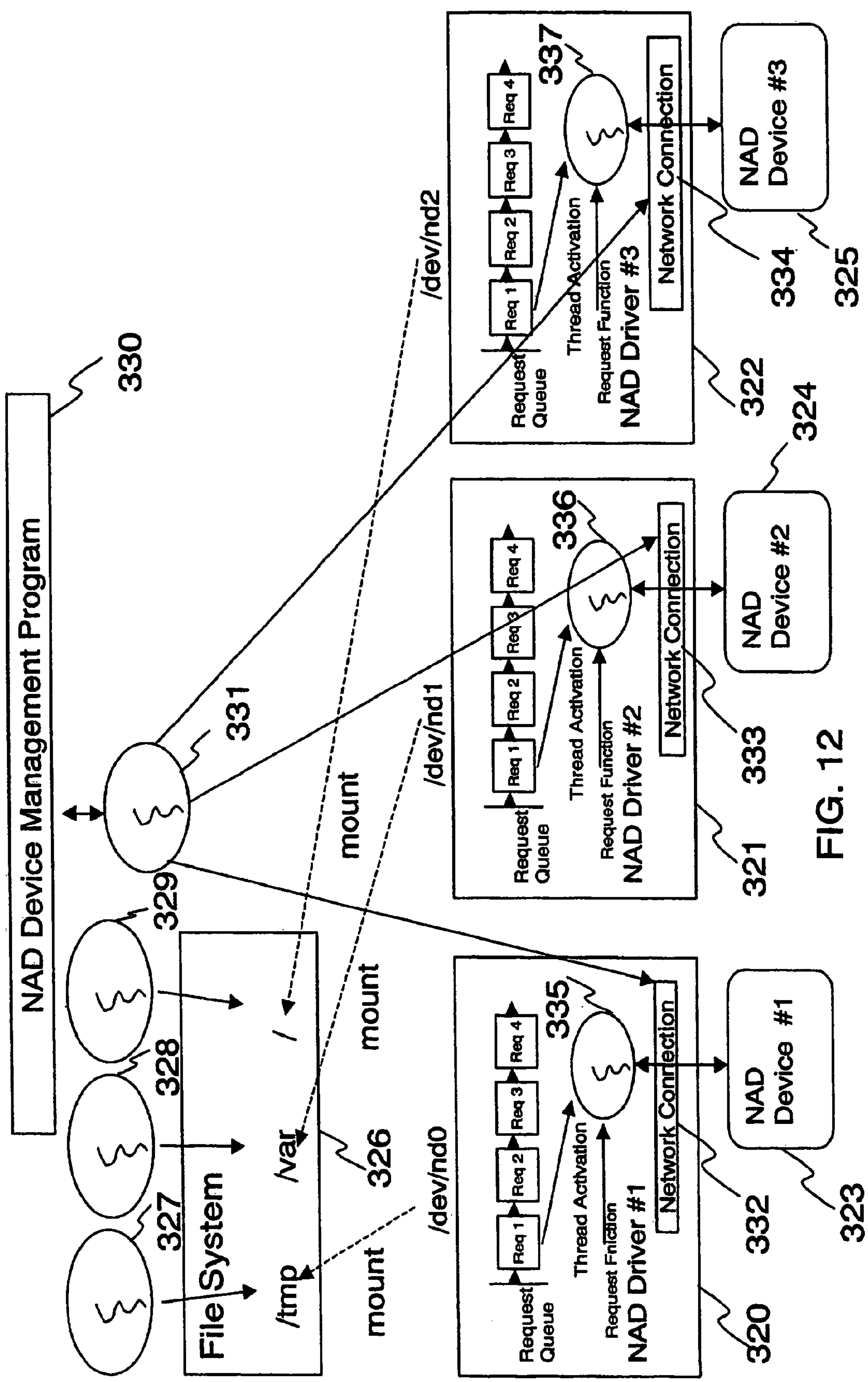

NAD Device Management Program
330
331
327
328
329
File System
/tmp
/var
/
326
mount
mount
mount
/dev/nd0
/dev/nd1
/dev/nd2
Request Queue
Req 1
Req 2
Req 3
Req 4
Thread Activation
Request Function
NAD Driver #1
NAD Driver #2
NAD Driver #3
Network Connection
335
336
337
NAD Device #1
NAD Device #2
NAD Device #3
320
321
322
332
333
334
323
324
325

FIG. 12

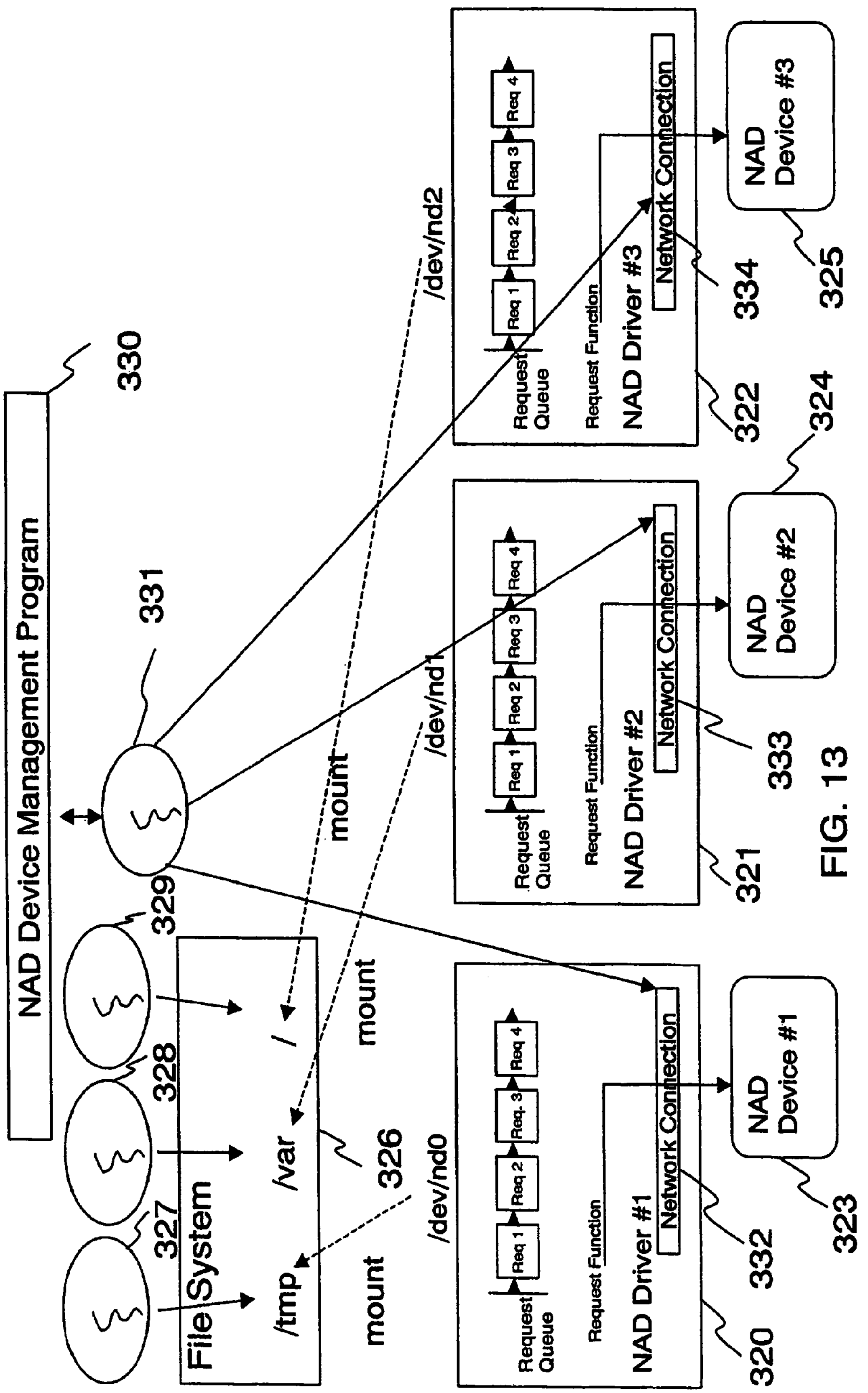

NAD Device Management Program
330
331
329
328
327
File System
/tmp
/var
/
326
mount
mount
mount
/dev/nd0
/dev/nd1
/dev/nd2
Request Queue
Req 1
Req 2
Req 3
Req 4
Req. 2
Req. 3
Request Function
NAD Driver #1
NAD Driver #2
NAD Driver #3
Network Connection
NAD Device #1
NAD Device #2
NAD Device #3
320
321
322
323
324
325
332
333
334

FIG. 13

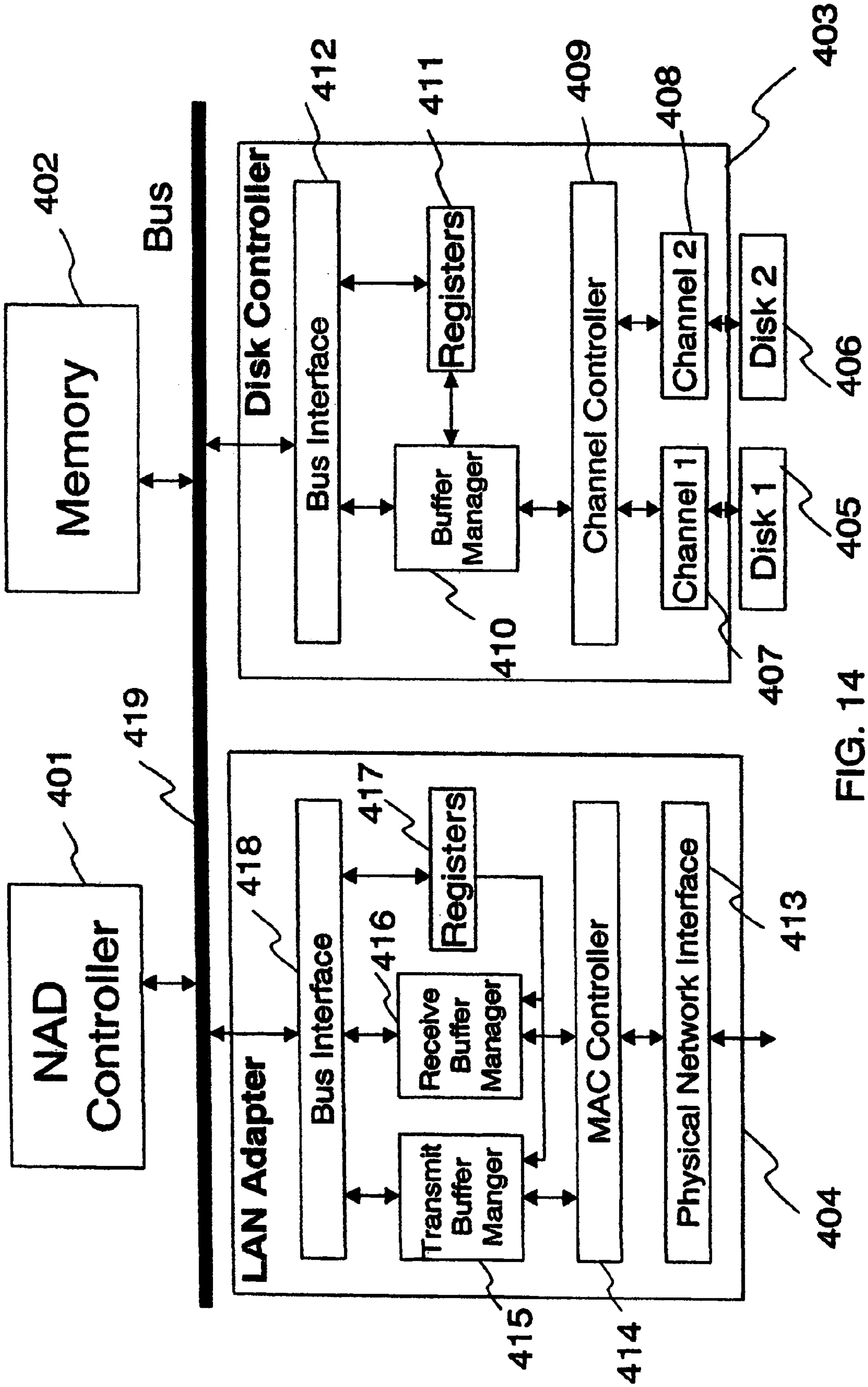

402
Memory
Bus
401
NAD Controller
419
Disk Controller
412
Bus Interface
411
Registers
Buffer Manager
410
409
Channel Controller
408
Channel 2
Channel 1
407
Disk 2
406
Disk 1
405
403
LAN Adapter
418
Bus Interface
417
Registers
416
Receive Buffer Manager
Transmit Buffer Manger
415
MAC Controller
414
Physical Network Interface
413
404

FIG. 14

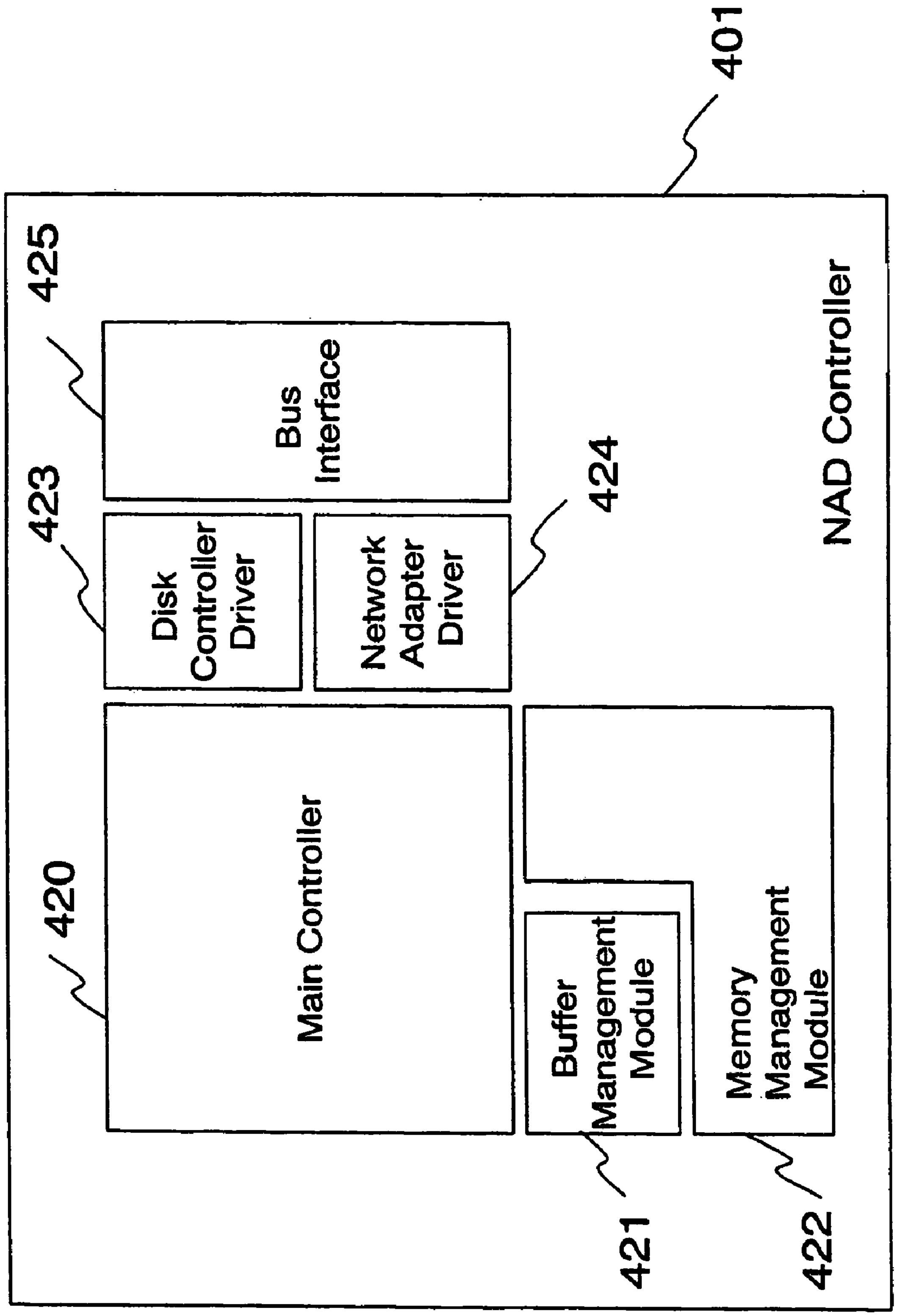

401
425
Bus Interface
423
Disk Controller Driver
Network Adapter Driver
424
NAD Controller
420
Main Controller
Buffer Management Module
Memory Management Module
421
422

FIG. 15

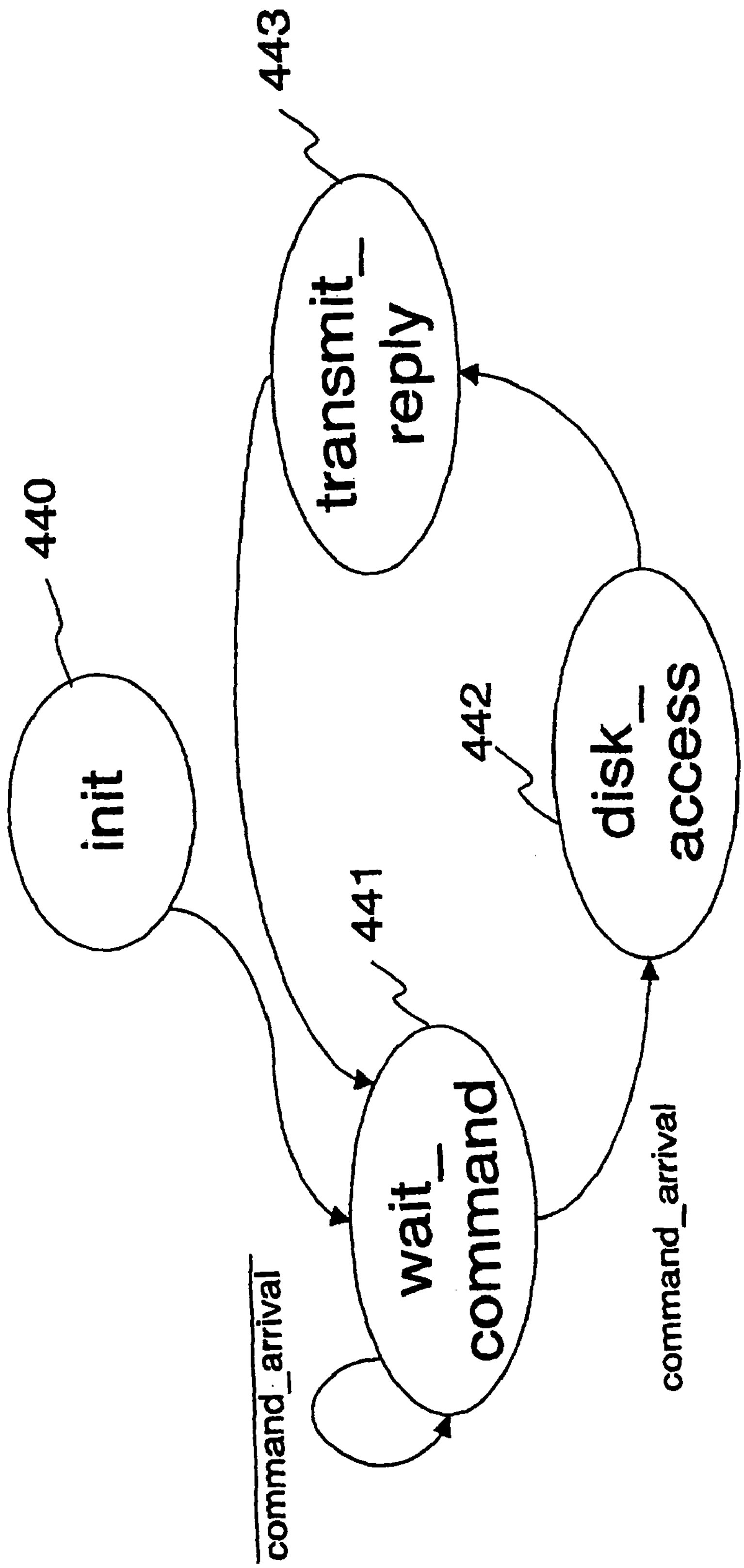

440
init
441
wait_
command
442
disk_
access
443
transmit_
reply
command_arrival
command_arrival

FIG. 16

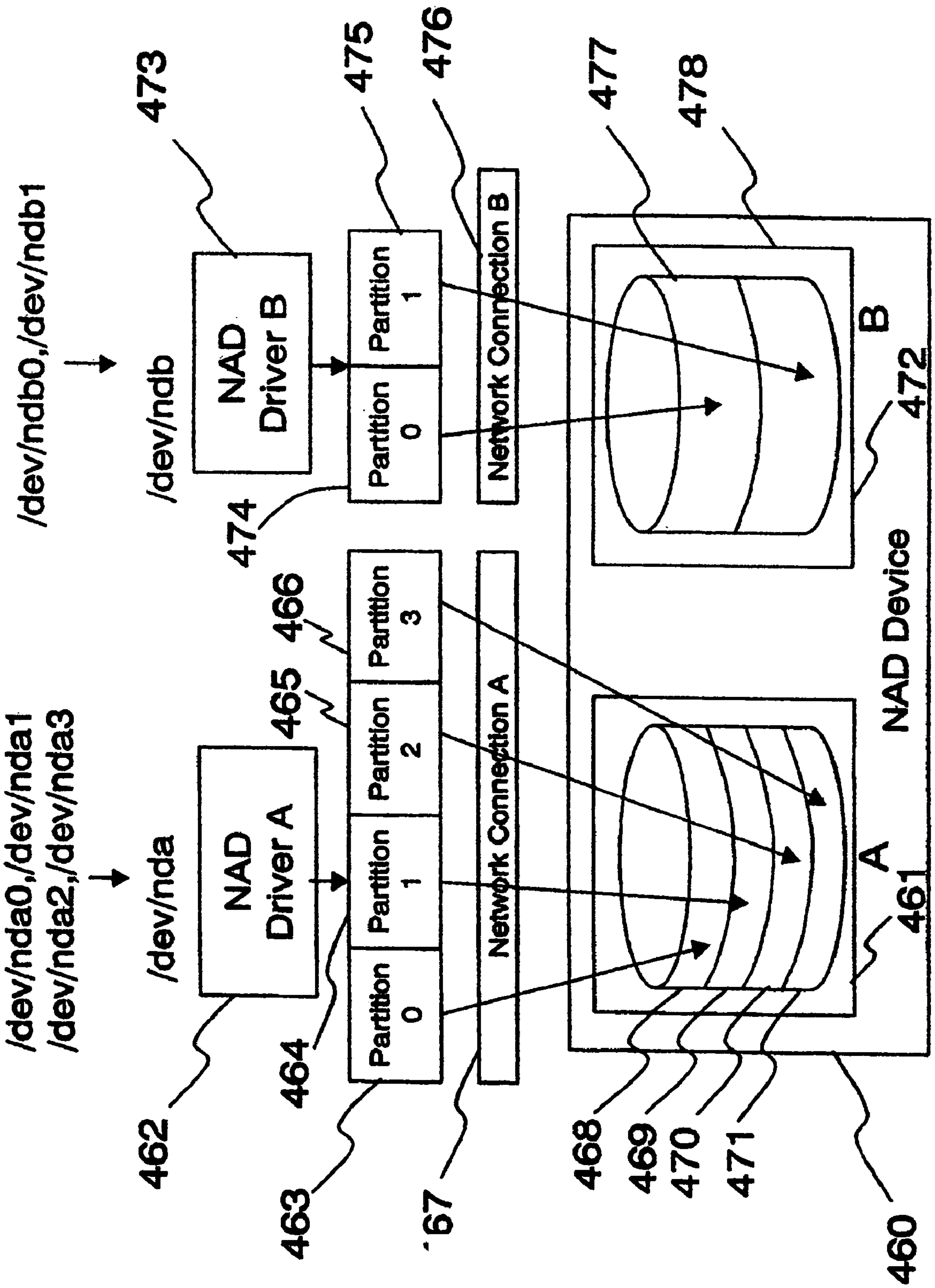

/dev/nda0,/dev/nda1
/dev/nda2,/dev/nda3
/dev/nda
NAD Driver A
/dev/ndb0,/dev/ndb1
/dev/ndb
NAD Driver B
Partition 0
Partition 1
Partition 2
Partition 3
Partition 0
Partition 1
Network Connection A
Network Connection B
A
B
NAD Device
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478

FIG. 17

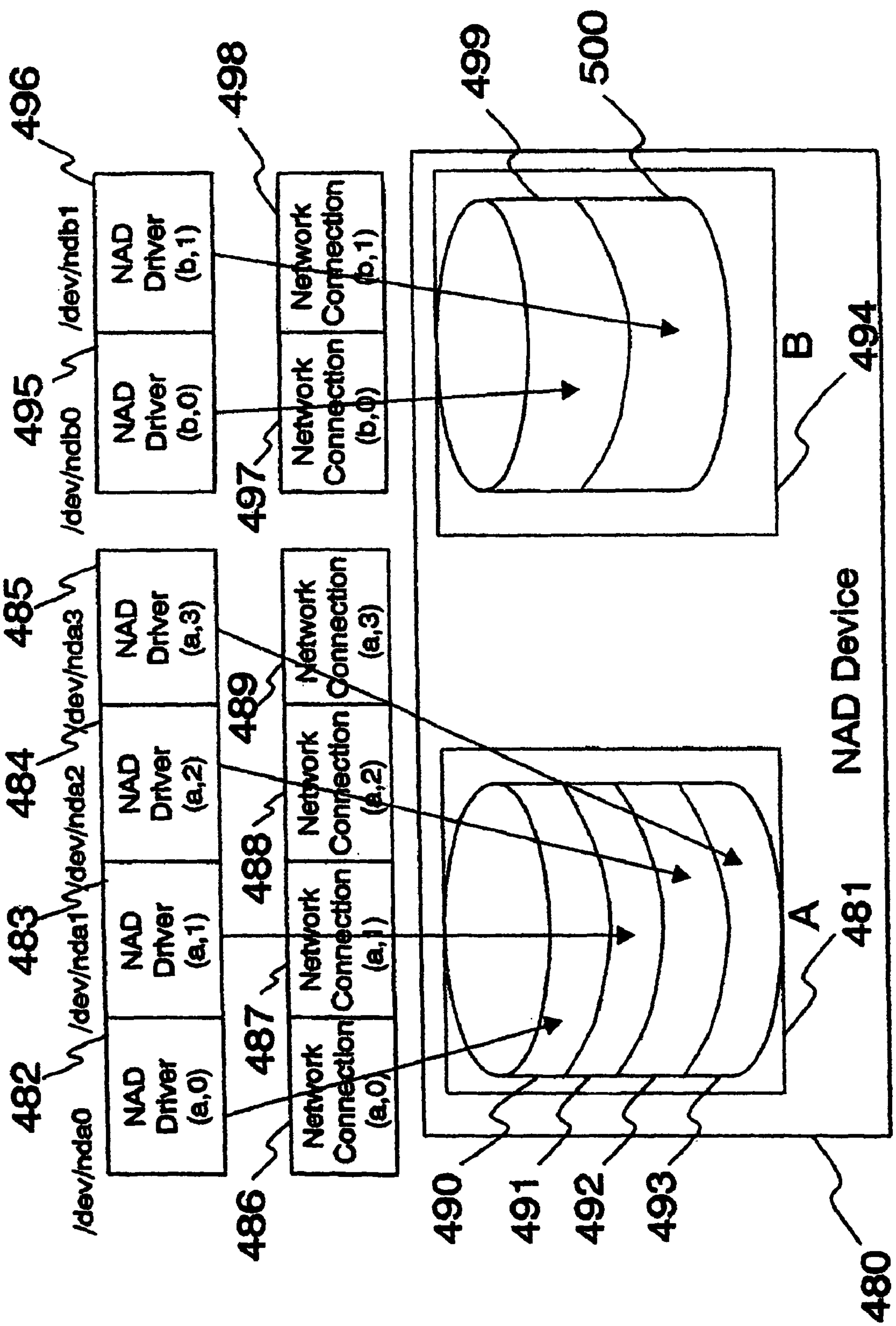
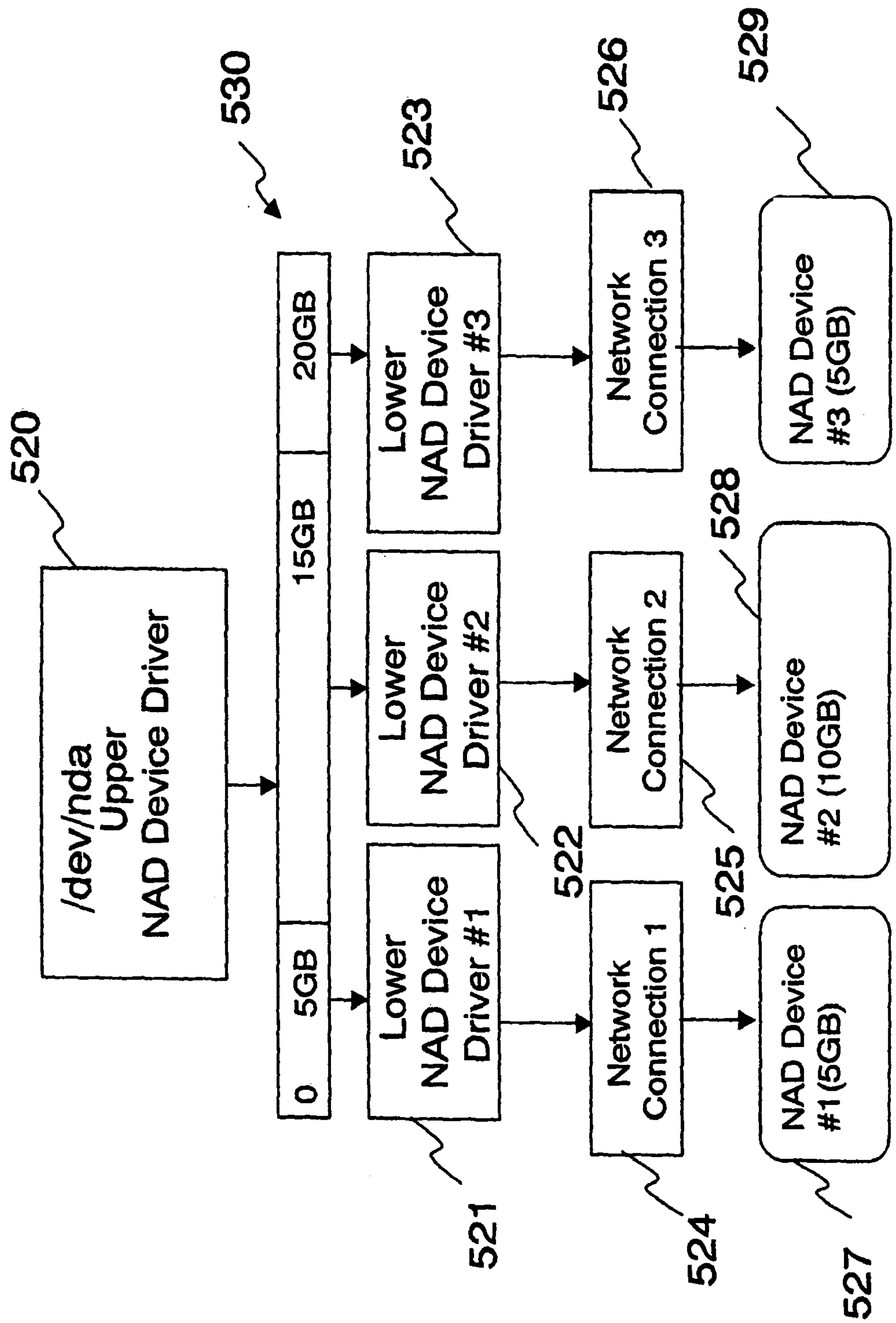

/dev/nda0
/dev/nda1
/dev/nda2
/dev/nda3
/dev/ndb0
/dev/ndb1
NAD Driver (a,0)
NAD Driver (a,1)
NAD Driver (a,2)
NAD Driver (a,3)
NAD Driver (b,0)
NAD Driver (b,1)
Network Connection (a,0)
Network Connection (a,1)
Network Connection (a,2)
Network Connection (a,3)
Network Connection (b,0)
Network Connection (b,1)
A
B
NAD Device
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500

FIG. 18



520
/dev/nda
Upper
NAD Device Driver
530
0
5GB
15GB
20GB
Lower
NAD Device
Driver #1
Lower
NAD Device
Driver #2
Lower
NAD Device
Driver #3
521
522
523
Network
Connection 1
Network
Connection 2
Network
Connection 3
524
525
526
NAD Device
#1 (5GB)
NAD Device
#2 (10GB)
NAD Device
#3 (5GB)
527
528
529

FIG. 19

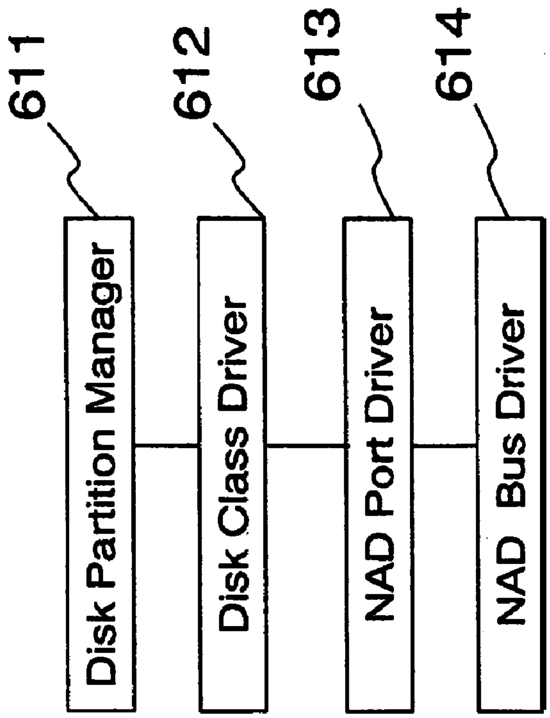


FIG. 20B

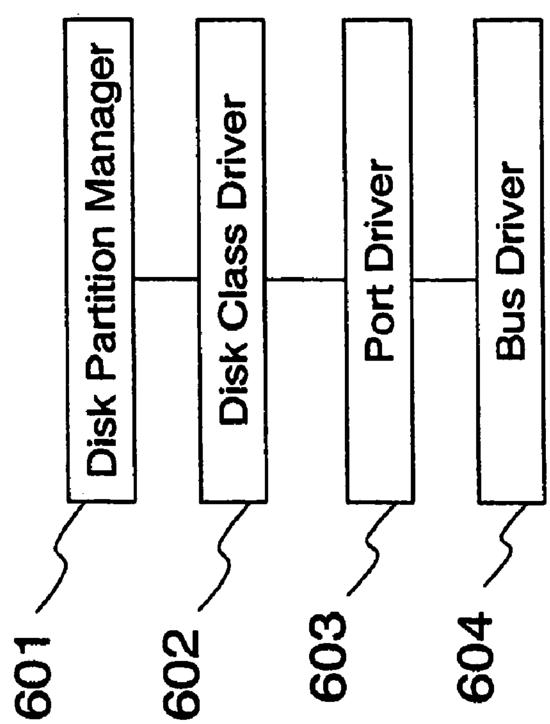


FIG. 20A

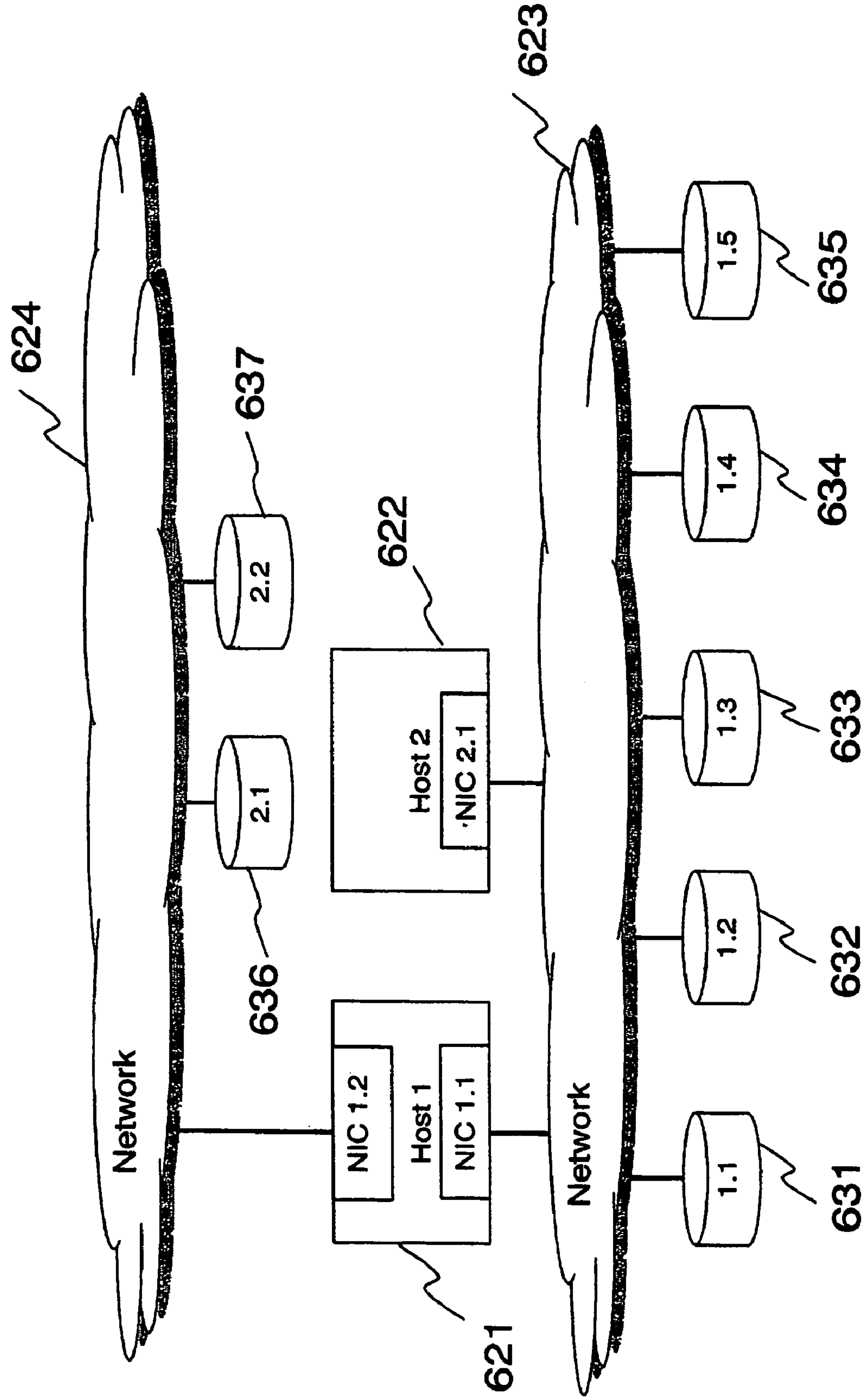

624
637
2.2
622
623
635
1.5
634
1.4
633
1.3
Host 2
NIC 2.1
2.1
636
632
1.2
Network
NIC 1.2
Host 1
NIC 1.1
Network
631
1.1
621

FIG. 21

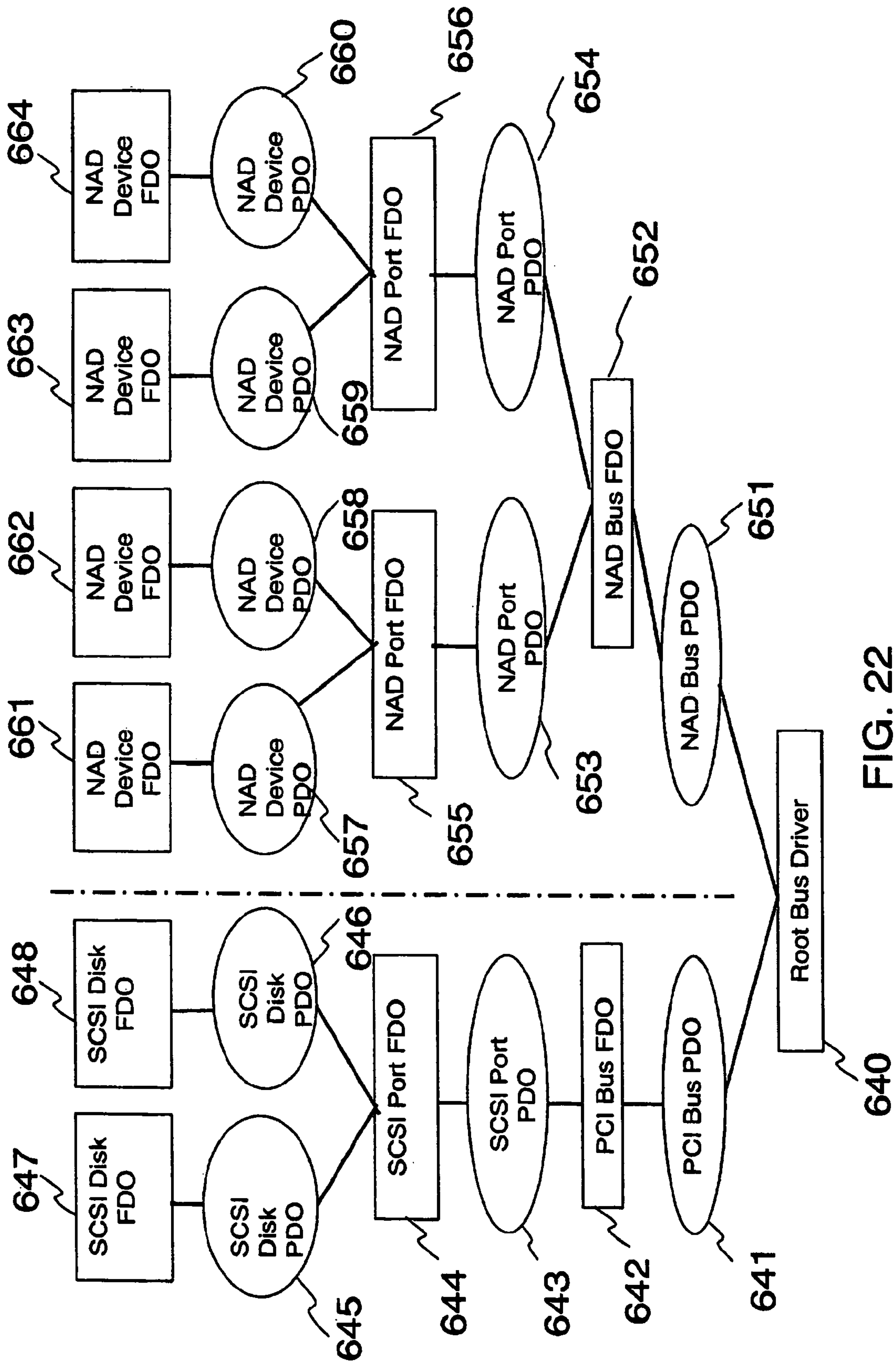

SCSI Disk FDO
647
SCSI Disk FDO
648
SCSI Disk PDO
645
SCSI Disk PDO
646
SCSI Port FDO
644
SCSI Port PDO
643
PCI Bus FDO
642
PCI Bus PDO
641
Root Bus Driver
640
NAD Device FDO
661
NAD Device FDO
662
NAD Device FDO
663
NAD Device FDO
664
NAD Device PDO
657
NAD Device PDO
658
NAD Device PDO
659
NAD Device PDO
660
NAD Port FDO
655
NAD Port FDO
656
NAD Port PDO
653
NAD Port PDO
654
NAD Bus FDO
652
NAD Bus PDO
651

FIG. 22

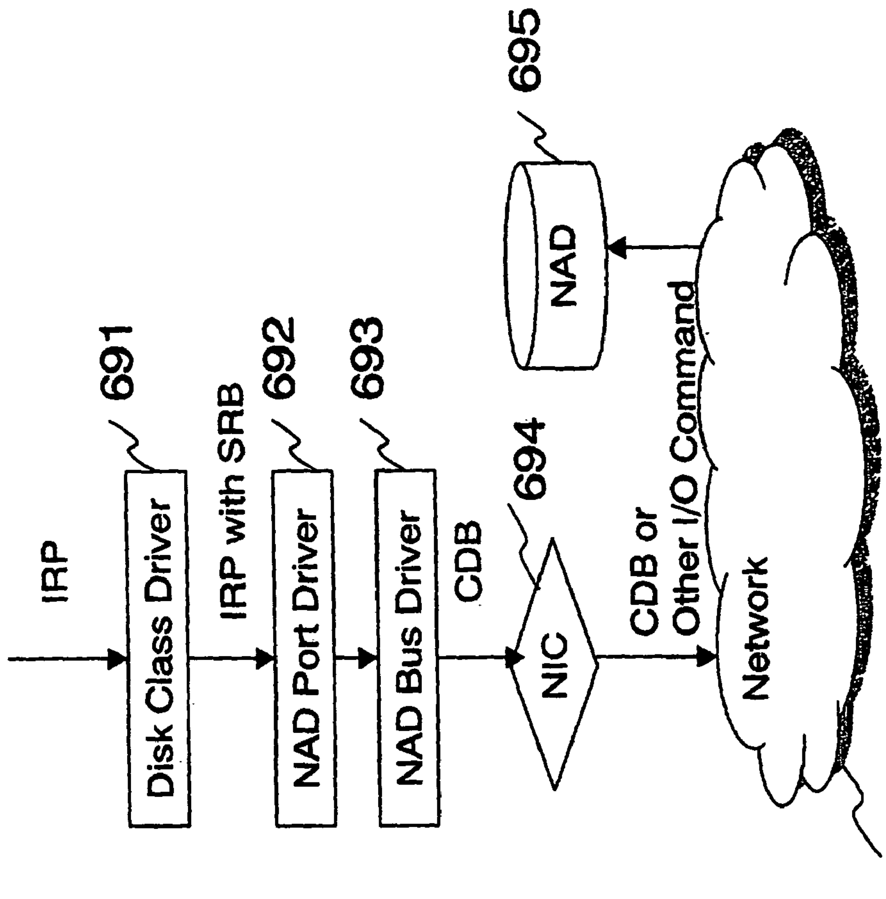


FIG. 23B

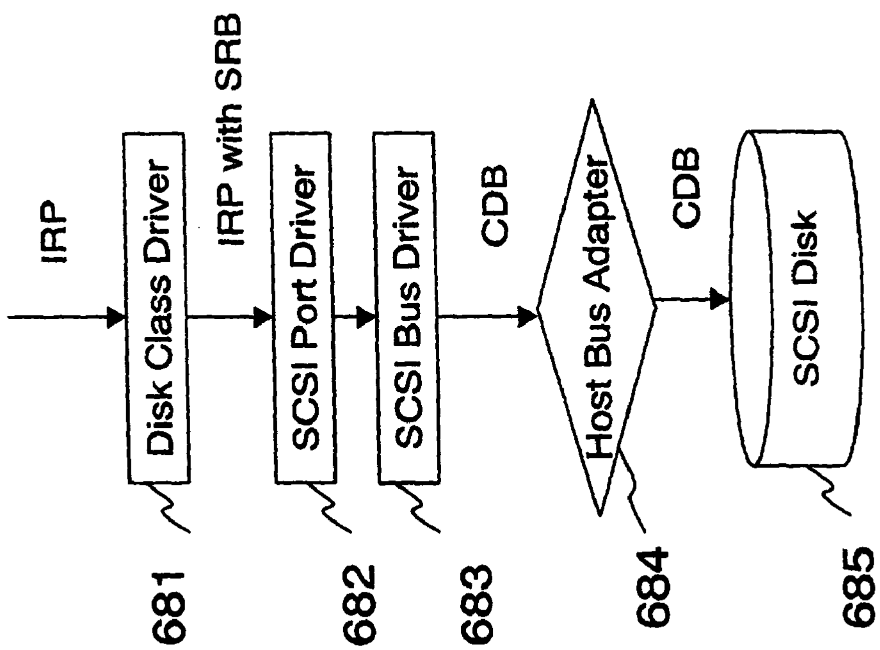


FIG. 23A

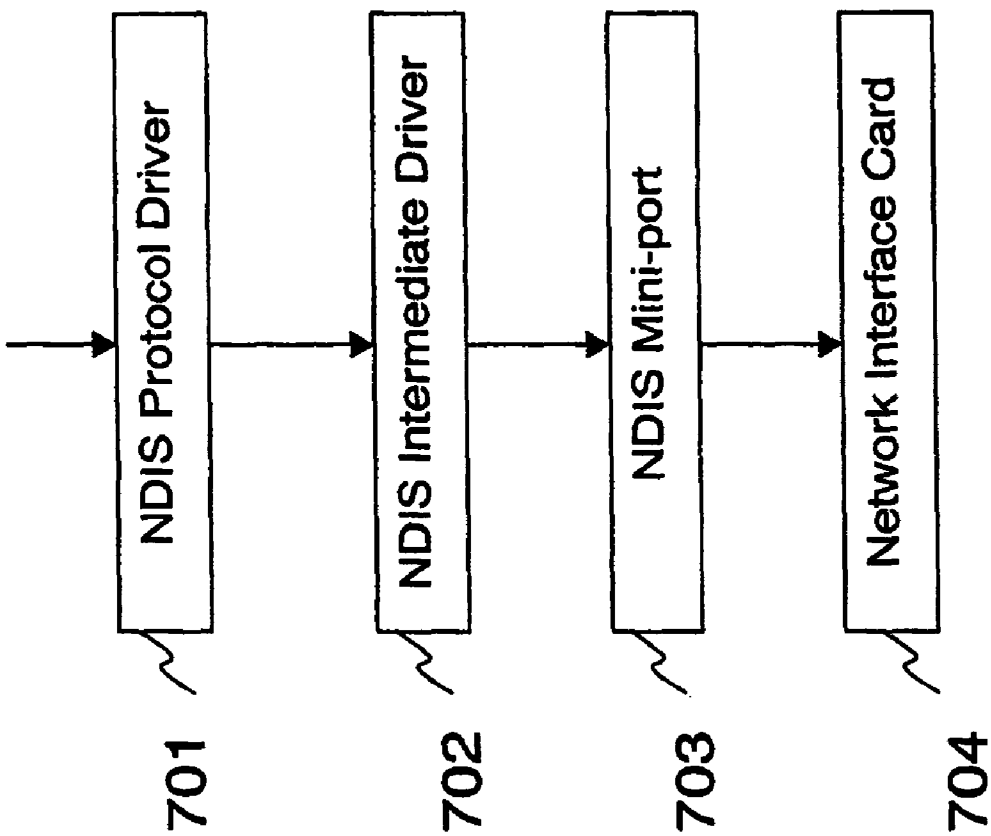

NDIS Protocol Driver
NDIS Intermediate Driver
NDIS Mini-port
Network Interface Card
701
702
703
704

FIG. 24

DISK SYSTEM ADAPTED TO BE DIRECTLY ATTACHED

RELATED APPLICATION

This application is a divisional patent application of U.S. patent application Ser. No. 09/974,082, filed on Oct. 9, 2001, and entitled "Disk System Adapted to Be Directly Attached"; which claims the benefit under 35 U.S.C. 119(e) to U.S. provisional patent application No. 60/240,344, filed Oct. 13, 2000, and entitled "Disk System Adapted to Be Directly Attached to Network"; the disclosures of which are hereby incorporated herein in their entirety.

BACKGROUND OF THE INVENTION

1. Technical Field

This invention in general relates to computer systems. More specifically, this invention relates to a disk system or interface that can be directly attached to a network.

2. Description of the Related Art

As the Internet becomes popular, the amount of data that needs to be stored has drastically increased. Especially, there is a high demand for a high-capacity disk drive to store multimedia data. For example, a demand for a disk system having a disk capacity of terabytes per server is not unusual.

A tape drive or a CD drive may be used to store such amount of data, but its performance and user convenience are not matched to those of a hard disk drive. However, increasing the capacity of a hard disk in a conventional server system presents some problems.

There are NAS (Network Attached Storage) products that can be connected to a network, usually Ethernet, to provide a pre-configured disk capacity along with integrated system/storage management using the NFS (Network File System) protocol, the CIFS (Common Internet File System) protocol, or both on top of the IP protocol used on the Internet. The primary purpose of these protocols is to exchange files between independently operating computers. Therefore, the client using the NAS for file access experiences the difference between its local storage and the storage in the NAS systems.

The NAS is basically a stripped-down version of a file server having mainly the functions of storing and retrieving files. Accordingly, increasing a disk capacity using a NAS product amounts to adding a separate file server in practice, which presents many shortcomings. Since an NAS disk is not seen as a local disk to the client, the installation, movement, and administration of an NAS disk should be done only through the operating system and software offered as part of the NAS system. An NAS disk is installed in the inside bus of the NAS system, leading to a limitation to the number of disks that can be installed. Since the NAS system has a hard disk under its own operating system, the client cannot use an arbitrary file system to access the hard disk. Further, the NAS system requires an IP address. Overall, not only the installation and administration costs per disk are more expensive than those of a local disk, but also user convenience is severely restricted.

There is SAN (Storage Area Network) that uses the Fibre Channel technology. To use the devices connected to a SAN, a special-type of switch is needed. For example, Fiber Channel uses a Fibre Channel hub or a Fibre Channel switch. The SAN has some shortcomings. Typically, a separate file server is used. In general, the SAN equipment is expensive, and so is the administration cost of the SAN system because, for example, it often requires an administrator with a specialized knowledge on the system.

Therefore, there is a need for an interface that allows a disk system to be directly attached to a network, while still being accessed like a local disk without the need of adding an additional file server or special equipment.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a disk system that can be directly attached to a network connecting to a host without going through a network file system.

Another object is to provide a disk system that can be recognized and used as a local disk to a host without requiring additional facility such as an additional file server, a special switch, or even an IP address, if appropriate.

Still another object of the present invention is to provide a disk system that can be conveniently connected to a server without much intervention of network/server administration.

Yet another object is to provide a low-cost disk system, many of which can be plugged into existing network ports to readily satisfy a disk capacity demand.

Further object is to provide an interface that allows a device attachable to a bus to be plugged into a network port.

The foregoing and other objects are accomplished by providing a network-attached disk (NAD) system that includes an NAD device for receiving a disk access command from a host through a network, a device driver at the host for controlling the NAD device through the network, where the device driver recognizes the NAD device as a local device. The host may run the UNIX or Windows family of operating systems. The NAD device includes a disk for storing data, a disk controller for controlling the disk, and a network adapter for receiving a disk access command from the host through a network port.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a block diagram of an environment where a network-attached disk (NAD) system of the present invention is used.

FIG. **2** is an illustration of how multiple NAD devices may be accessed by multiple hosts through a network.

FIG. **3** is an example of how multiple NAD devices are accessed by multiple hosts.

FIG. **4** is another example of how each disk inside an NAD may be treated as a separate disk.

FIG. **5** is an illustration of how a block device driver, specifically an NAD device driver, is registered and unregistered under the UNIX operating system.

FIG. **6** is an illustration of the relation among the directory, device file, device driver, and device.

FIG. **7** is an illustration of how a request function directly issues a command to a device.

FIG. **8** is an illustration of how a request function activates a device accessing thread.

FIG. **9** is a block diagram of a local disk system and that of an NAD device running under UNIX.

FIG. **10** is an illustration of a device searching thread for identifying the attached NAD devices and for providing the NAD information to the NAD device management program.

FIGS. **11**A and **11**B are examples of network connections made between an NAD device driver and its corresponding NAD device using a connection setting thread.

FIG. **12** is an illustration of a method of implementing an NAD device driver, using a device accessing thread.

FIG. **13** is an illustration of a method of implementing an NAD device driver, without using a device accessing thread.

FIG. **14** is an example of an NAD device construction.

FIG. **15** is a functional block diagram of an NAD controller.

FIG. **16** is a simplified state transition diagram of a state machine used by the main controller of an NAD controller.

FIG. **17** is an illustration of how a disk inside an NAD device may be divided into disk partitions to which a device driver is assigned.

FIG. **18** is an illustration of how separate NAD device drivers may be generated so that a physically single disk can be assessed by different file systems.

FIG. **19** is an illustration of how the NAD system can recognize physically separate, several NAD disks as a logically single disk.

FIGS. **20**A and **20**B are illustrations of the hierarchies of the disk driver layers in the conventional disk system and the NAD system under the Windows 2000 operating system.

FIG. **21** is an illustration of a network environment where the NAD system of the present invention is used in the Windows 2000 operating system.

FIG. **22** is an example of a device stack created in the Windows 2000 operating system.

FIG. **23**A is an illustration of the flow of IRP, SRB, and CDB in a conventional disk system in the Windows 2000 operating system.

FIG. **23**B is an illustration of the flow of IRP, SRB and CDB in an NAD system in the Windows 2000 operating system.

FIG. **24** is an illustration of NDIS (Network Device Interface Specification) in the Windows 2000 operating system.

DETAILED DESCRIPTION OF THE INVENTION

FIG. **1** shows an environment where the present invention is used. A host **100** has a file system **101**, which may contain a local disk device driver **102** that controls a local disk **104** connected to an internal system bus **103**. A local device is defined as a device inside a standard-alone system as opposed to a network device connected to a network. Local devices are directly connected to a system bus often through an adapter called a host bus adapter allowing the host to communicate with the devices without going through any network, whereas network devices are not directly connected to a system bus, rather connected through an interface called a network interface card (NIC) installed on system bus. The local disk **104** may be a conventional IDE (Integrated Drive Electronics) disk or SCSI (Small Computer System Interface) disk.

The file system **101** also contains a network-attached disk (NAD) device driver **105** of the present invention that controls an NAD device **108** connected through a network adapter device driver **106** and a network **107** such as Ethernet. The NAD device **108** of the present invention contains one or more disks **109**. The network **107** is an existing general-purpose network for carrying storage traffic as well as other application traffic. This so called "front-end" network for carrying general-purpose network traffic is distinguished from a "back-end" network dedicated to storage such as that used in the conventional Storage Area Network (SAN) scheme.

The present invention features two main components: one is the NAD device driver **105** at the host and the other is the NAD device **108** attached to the network.

FIG. **2** shows an example of how multiple NAD devices are accessed by multiple hosts through a network. NAD device #**1** **123** with disk(**1**,**1**) **126** and NAD device #**2** **124** with disk(**2**,**1**) **127** and disk(**2**,**2**) **128** are accessed by Host #**1** **120** through a network **122**, while NAD device #**3** **125** with disk (**3**,**1**) **129**, disk(**3**,**2**) **130**, disk(**3**,**3**) **131** is accessed by Host #**2** **121** through the same network **122**.

Each disk appears to the host as if it is a local disk to connected to the system bus of the host so that each disk can be dynamically installed or removed. The present invention achieves this by creating a virtual host bus adapter in purely software means that recognizes an NAD device as if it is connected to the system bus although there is no physical host bus adapter connected the NAD. This is distinguished from the conventional Network Area Storage (NAS) scheme where a NAS device connected through the NIC is still recognized as an independent file server connected to a network.

The Open Systems Interface (OSI) model defines 7 layers of protocols: a physical layer for electrical interface definitions, a data link layer for communication using data frames, a network layer for routing packets from one end to another, a transport layer for dividing messages into packets, a session layer for establishing communication session, a presentation layer for data presentation format, and an application layer for application programs. The present invention uses a data link layer protocol to contain storage commands into data link frames. Because the NAD device is not acting as an independent devices to the host, there is no need to use a network address such as IP address.

Since the specific configuration of the hosts and the disk systems can be dynamically changed, user convenience and portability is preserved as in the case of using a local disk. There is virtually no restriction to the number of disk systems that can be attached to the network, thus providing an unlimited disk storage capacity for a host.

FIG. **3** shows another example of how multiple NAD devices are accessed by multiple hosts through a network. NAD device #**1** **143**, NAD device #**2** **145**, and NAD device #**5** **147** are accessed by Host #**1** **140** through a network **142**, while NAD device #**2** **144** and NAD device #**4** **146** are accessed by Host #**2** **141** through the same network **142**.

The disks contained in an NAD may be treated as separate disks so that each of them can be independently accessed by a host. FIG. **4** shows an example of treating each disk inside an NAD device as separate disks. Disk(**1**,**1**) **166** inside NAD device #**1** **163**, disk(**2**,**2**) **168** inside NAD device #**2**, and disk(**3**,**2**) **170** inside NAD device #**2** are accessed by Host #**1** **160** through a network **122** while disk(**2**,**1**) **167** inside NAD device #**2** **164** and disk (**3**,**1**) **169** and disk (**3**,**3**) **171** inside NAD device #**3** **165** are accessed by Host #**2** **161** through the same network **162**. Note that disk (**2**,**1**) **167** and disk (**2**,**2**) **168**, inside NAD device #**2** **164**, are independently accessed by Host #**1** **160** and Host #**2** **161** respectively.

Block Device Driver

An embodiment of the NAD system will be explained with an example running the UNIX family of operating systems although other operating systems such as Windows may also be used.

Each block device for block data storage, such as a disk device, is assigned a major device number to distinguish among different kinds of block devices, and a minor device number to distinguish among same kinds of block devices. In UNIX, each device is accessed through a device file, which provides an interface for accessing the real device. Device files are usually generated in advance, each with a major device number and a minor device number as well as information on a block device driver.

The purpose of the device driver is to handle the requests made by the kernel with respect to a device. The device driver isolates device-specific codes to provide a consistent interface for the kernel. In order to activate the operation of a device driver, a device file and device driver routines must be prepared, after which the functions of the driver routines must be registered so that the operating system such as UNIX can understand their availability. This is usually done by passing the major number assigned to the device and the functions of the driver routines as parameters.

Registration and Unregistration of Block Device Driver

Once a block device driver is registered by passing the device's major device number and the driver functions as parameters, it may be unregistered by passing the major number.

Table 1 lists the functions used to either register or deregister a device driver.

TABLE 1

Functions	Description
Register__blkdev( )	register a driver by taking a major number and driver functions as parameters
Unregister__blkdev( )	unregister a driver by taking a major number

Table 2 lists the general functions used by the local driver and the NDA driver.

TABLE 2

Driver Function	Description
Read( )	used to read data in the device
write( )	used to write data in the device
ioctl( )	used to change a particular value of a structure for a driver or to control input/output with respect to a device
open( )	used to initialize a driver
release( )	used to eliminate a driver
fsync( )	used to reflect the content of buffer cache to the real device driver
check__media__change( )	used to sense a change in the device condition
revalidate( )	used to update device managed by the deriver and device driver itself

Table 3 lists examples of the driver functions specific to the IDE local disk driver and the NAD driver.

Use of Block Device

Once a block device driver is registered and its device file is generated, read/write is done to the device file to access the real device. The device file, however, is not directly called by the user, rather called after being mounted to the file system. Before being mounted, each block device file must be formatted according to a particular file system. Since the NAD device driver of the present invention is prepared in the same way as a conventional local disk driver, the set of I/O commands used to format a conventional local disk can also be used to format a disk in the NAD device. In addition, since NAD devices are controlled in the device driver level, they can be formatted in a required format independent of the file system used.

FIG. **6** shows the relation among the directory, device file, device driver, and device. The left side shows attachment of a conventional local disk system where a device file **201** mounted on a directory **200** is used by a local disk device driver **202** to control a local disk **203**. The right side shows an NAD system of the present invention where a device file **204** mounted on the directory **200** is used by an NAD device driver **205** to control a NAD device **207** through a local area network (LAN) **206** such as Ethernet. The two relations are similar except that the NAD device is accessed through the network.

Structure of Block Device Driver

Each block device driver has an I/O request queue to store the I/O requests to the device. The stored requests may be re-scheduled for the purpose of improving the performance. Besides the I/O request queue, each block device driver needs a request function to process the I/O requests in the queue.

FIG. **7** shows a situation where the request function directly issues a command to a block device. An NAD device driver **220**, using a device **222** and a file system **223**, has a queue **224** that stores I/O requests **225** through **228**. The NAD device driver **220** has a request function **229** that issues a command to the NAD device **221** by taking a currently processed request **225**.

TABLE 3

Driver Function	IDE Local Driver Function	NAD Driver Function
read( )	ide__read( )	netdisk__read( )
write( )	ide__write( )	netdisk__write( )
ioctl( )	ide__ioctl( )	netdisk__ioctl( )
open( )	ide__open( )	netdisk__open( )
Release( )	ide__release( )	netdisk__release( )
fsync( )	ide__fsync( )	netdisk__fsync( )
check__media__change( )	ide__check__media__change( )	netdisk__check__media__change( )
revalidate( )	ide__revalidate( )	netdisk__ide__revalidate( )

FIG. **5** shows an example where a block device driver, specifically an NAD device driver, is registered and unregistered. Initially, an IDE device driver **181** with major device #**3** is created as well as some null device drivers such as major device #**0** **180** and major device #n **182**. Major device #**60**, **183**, the NAD device driver that is assigned a major device number of **60**, is registered by using a device registration function of register_blkdev(**60**, fops) **185**. Later the NAD device driver is deregistered into major device #**60** **184**, a null device driver, by using a device de-registration function of unregister_blkdev(**60**) **186**. The figure shows that NAD device driver of the present invention is installed in the same way as the existing block device drivers.

FIG. **8** shows a situation where the request function **229**, instead of directly issuing a command, activates a device accessing thread **230** so that the device accessing thread **230** can issue a command based on the information in the request queue. A thread refers to a single use of a program that can handle multiple users.

Constitution of Local Disk System and NAD System

FIG. **9** shows the constitution of a conventional local disk system and that of the NAD system operating under UNIX. Under a file system **260**, a conventional local disk **264** attached to a local bus **263** is accessed by a conventional local disk device driver **261** through a bus interface **262**. Under the same file system **260**, an NAD device **272** with a disk **273**, attached to a network **271**, is accessed by an NAD device driver **265** through a network interface including a network protocol stack **266**, a network adapter device driver **267**, a bus interface **268**, and a network adapter **270**.

Since an NAD device is to be used like a local disk, the conventional local disk system and an NAD system of the present invention share a basic structure. The difference is that since an NAD system must communicate with an NAD device through a network, a protocol stack is added for network communication. The NAD driver delivers an I/O command to an NAD device through a network adapter and receives a response from the NAD device.

When an NAD device is accessed, either DMA (Direct Memory Access) or PIO (Programmed Input/Output) may be used. A conventional disk device driver operates in a DMA mode by issuing a DMA I/O command to a local disk with a starting buffer address and a byte transfer count. The local disk then takes over the data transfer, after which it interrupts the CPU. Similarly, the NAD device driver may be implemented to operate in a DMA mode by having the NAD device driver deliver an I/O command to an NAD device, which then completes the data transfer, after which it interrupts the CPU.

The conventional disk driver operates in a PIO mode by the CPU transferring data directly through data registers of the disk device until a particular data block is processed. Similarly, the NAD driver may be implemented to operate in a PIO mode by having the NAD device driver deliver a command to an NAD device and continue to transmit/receive data until a particular block of data is processed.

The network protocol that can be used in the present invention is not restricted to a particular protocol. It can be any connection-oriented protocol including TCP/IP. A connection-oriented protocol ensures that packets are not lost and packets are received in the order they are transmitted. If TCP/IP is used, an IP address must be used for each NAD device.

Local Disk Driver and Generation of NAD Driver

Once UNIX starts, if hardware scan detects any conventional local disks, their corresponding drivers are generated according to the units of the local disks or according to the units of disk partitions. In a similar fashion, NAD devices are identified during initial hardware scan and their corresponding drivers acting as a virtual host bus adapter must be generated. The drivers may be generated automatically by using a device searching thread that periodically identifies NAD devices attached to the network or manually by a system administrator using an NAD management program.

FIG. **10** shows a device searching thread for identifying the attached NAD devices and for providing the NAD device information to an NAD device management program. A thread **280** is run in a host **290** through a network protocol stack **282** and a network adapter driver **283** to identify NAD devices **285** through **289** together with the size and device file of each NAD to provide the information **281** to NAD device management program. Once informed of NAD device files available, the user then mounts a selected NAD device file to use a particular NAD device as a local disk.

Network Connection Between NAD Device Driver and NAD Device

In a conventional local disk, disk I/O is performed by reading/writing to I/O ports of the disk controller attached to the internal system bus. But the NAD device driver performs I/O to the corresponding NAD device through a network link. Instead of read/write to an I/O port, I/O is performed by read/write to a network connection such as a socket in UNIX. Therefore, a network connection such as a UNIX socket must be set up between the NAD device driver and NAD device.

FIGS. **11**A and **11**B show examples of network connections between an NAD device driver and the corresponding NAD device using a connection setting thread. NAD device #**1 302** is connected to NAD device driver #**1 301** through a network connection #**1 303** created by the ioctl( ) function, **304** while NAD device #**2 307** is connected to NAD device driver #**2 306** through a network connection #**2 308** created by the ioctl( ) function **309**.

Implementation of NAD Driver

FIG. **12** and FIG. **13** show two methods of implementing an NAD device driver, the former with a device accessing thread, and the latter without a device accessing thread.

FIG. **12** shows three NAD drivers **320**, **321** and **322** with the device files of "/dev/nd0", "/dev/nd1", "/dev/nd2" to access NAD device #**1 323**, NAD device #**2 324**, and NAD device #**3 325**, respectively. Each device file is mounted to "/temp", "/var", "/" directory in the file system **326**, respectively. User threads **327**, **328** and **329** for accessing the file may read/write on the NAD device through the file system **326**. A connection setting thread **331** provides the list of NAD devices available to an NAD device management program **330**. Based on the user's input, the connection setting thread **331** creates network connections **332**, **333** and **334**, as necessary.

Referring to FIG. **12**, when the user thread requests a file through a file system, the file system first checks the buffer cache to find out whether the requested file block is in the buffer. If the block is in the buffer, the user thread refers to the block. But if the block is not in the buffer, data must be read from the NAD device. The user thread puts the request on the request queue, activates an NAD accessing thread **335** (or **336**, **337**) responsible for NAD device control through a request function, and the user thread blocks itself. The user thread blocked is awakened later by the NAD accessing thread, such as **335**, that received the corresponding data.

FIG. **13** is similar to FIG. **12** except that the user thread now directly requests data from the NAD device rather than using an NAD accessing thread. For example, the user thread puts the request on the request queue, activates a software interrupt that will actually handle block data transfer between the NAD device and the host, and the user thread blocks itself. Once the data transfer is done, an interrupt is generated to wake up the blocked thread.

Communication Protocol Between Host and NAD Device

When a host NAD device driver accesses an NAD device for I/O, the position of the first block and the number of blocks are given as parameters of the I/O command. Or, in the case of SCSI, the I/O command may be in the form of a CDB (Command Descriptor Block).

To transfer the CDB or the block transfer information, a reliable communication protocol is necessary. The present invention is not limited to a particular kind of communication protocol as long as a connection-oriented protocol is used including TCP/IP. A connection-oriented protocol means that packets can be retransmitted in the case packets are lost, and received packets are arranged at the receiver end in the order they were sent.

NAD Device

FIG. **14** shows a functional block diagram of the NAD device of the present invention. A preferred embodiment of the NAD device is comprised of an NAD controller **401** for controlling the whole NAD device, memory **402**, a disk controller **403** for executing a disk access command, one or more disks **405**, **406**, and a LAN adapter **403** for receiving a disk access command from a host through a network. The NAD controller **401**, the memory **402**, the disk controller **403**, and the LAN adapter **404** are all connected to a bus **419** internal to the NAD device.

The disk controller **403** is a module that performs disk I/O operations by controlling the disks **405** and **406** over internal disk channels. The disk controller **403** is further comprised of one or more disk channels **407** and **408** controlled by a channel controller **409**, a buffer manager **410**, some registers **411**, and a bus interface **412**. The buffer manager **410** consults the registers **411** to obtain a disk sector number and a channel to execute a disk access command. The buffer manager **410** also commands the channel controller **409** to transfer data from the memory to disk channel **407** or **408** or vice versa as a result of executing a disk access command. The channel controller **409** actually accesses the disk over the disk channel **407**, **408** to transfer data from the disk to the memory or vice versa.

The LAN adapter **404** is a module that receives disk I/O command packets from the host and transmits replay packets over the network. The LAN adapter **404** is further comprised of a physical network interface **413** for interfacing with the network, a MAC (media access control) controller **414**, transmit buffer **415** for storing transmit data, a receive buffer **416** for storing receiving data **416**, registers **417**, and a bus interface **418**.

The bus interface **418** transfers data from the bus to the transmit buffer **415**, the receive buffer **416**, and the registers **417**, or vice versa. The MAC controller **414** transfers data to the physical network interface **413** so that the physical network interface can transmit the data to the host computer. When the physical network interface **413** receives a disk I/O request packet from the host computer, it transfers the packet to the MAC controller **414** so that the MAC controller can extract necessary data from the packet and transfer the data to the receive buffer **416**.

FIG. **15** shows that the NAD controller **401** may be comprised of a main control **420** for controlling the NAD, a buffer management module for caching data in the disk **421**, a memory management module for managing assignment of memory space **422**, a disk controller driver **423** for interfacing with the disk controller, a network adapter driver **424** for interfacing with the network adapter, and a bus interface **425** for interfacing with the bus inside the NAD.

The NAD controller **401** mainly executes I/O commands from the host's NAD device driver, but it can perform other additional functions. For example, a filter program can be installed to NAD so as to provide features that are not offered in the host, for example, a back up operation. Other examples include access control, access share, access right transfer, etc. Specifically, a filter program can be installed to limit access to an NAD device to a certain time period, to allow several hosts simultaneously access an single NAD, or to transfer one host's access rights to another host. The NAD device driver at the host can request to execute the filter program at the time of I/O command execution through the ioctl( ) function in UNIX.

FIG. **16** shows a simplified state transition diagram of a state machine used by the main controller **420**. At the 'init' state **440**, the state machine initializes all the NAD hardware and allocates memory for the disk controller **403** and the LAN adapter **404**. Upon completing the initialization process, the state machine makes a transition to 'wait-command' state **441** where the NAD system waits for an incoming I/O command from the host computer over the network. When such I/O command is received from the host computer, the state becomes the 'disk_access' state **442** where an appropriate disk I/O operation is performed through the disk controller. Upon completing the disk I/O, the state moves to the 'transmit_reply' state **443** where the NAD device sends the result to the host computer through the LAN adapter **404**. A person skilled in the art would appreciate that the state machine can be readily realized with a CPU and memory.

Network Adapter Driver and Disk Controller Driver

The network adapter and the disk control driver can be implemented at least in two ways. One uses an interrupt mechanism through DMA (Direct Memory Access) and the other uses polling through PIO (Programmable I/O). The former has the advantage of easy programming so that other jobs can be executed without a complete disposition of disk controller data. The latter has the advantage of dispensing with time delay due to interrupts, but has the disadvantage of an inefficient processor usage due to the time spent for continuous read and write.

Additional Embodiments of NAD Drivers

Usually, an NAD device driver is generated for each disk unit attached. However, just as a local disk may be partitioned, the disk inside an NAD device may also be partitioned into several disk partitions where each disk partition can be accessed by a separate device driver. Alternatively, several disks located in the physically separate NAD devices may be combined for use as a logically single disk.

FIG. **17** shows an example where the disk inside an NAD device may be divided into several disk partitions where all of the partitions are assigned a single device driver. An NAD driver A **462**, for example, is assigned to four partitions **463**-**466** so that the NAD driver A **462** refers the partition table in order to handle I/O requests directed to specific partitions **468** through **471** of a disk **461** inside a NAD device **460**, respectively, using a same network connection **467**. Similarly, an NAD driver B **473** is assigned to two partitions **474** and **475** so that the NAD driver B **473** can be used to control two disk partitions **477** and **478** of a disk **472** inside the NAD device **460**.

FIG. **18** shows an example where separate NAD device drivers may be generated so that a physically single disk can be assessed by different file systems. Disk A **481** inside an NAD device **480** is divided into four partitions **490** through **493**, and four separate NAD driver(a,**0**) through driver (a,**3**) **482** through **485** are created so that each NAD driver can control each disk partition through a separate network connection **486** (**487**, **488**, or **489**). Similarly, disk B **494** inside the NAD device **480** is divided into two partitions **499** and **500**, and two separate NAD drivers(b,**0**) and NAD drive (b,**1**) **495** and **496** are created so that each NAD driver can control each disk partition through a separate network connection **497** (or **498**). Since different network connections are used, this configuration enables a physically single hard disk to be mounted to different file systems.

FIG. **19** shows an example of how the present invention can recognize physically separate, multiple disks in different NAD devices as a logically single disk. Specifically, FIG. **19** shows that three lower-level NAD device drivers **521**, **522** and **523** controlling NAD device #**1** **527** of 5 GB, NAD device #**2** **528** of 10 GB, and NAD device #**3** **529** of 5 GB, respectively, through separate network connections **524**, **525** and **526**, are united into a single upper-level NAD driver **520** partitioned into a configuration **530**. The file system mounts "/dev/nda" to access the total space of 20 GB.

NAD System Running Under Windows Operating System

The foregoing system and method explained using examples running under the UNIX family of operating systems can equally be applied to implement an NAD system running under the Windows™ family of operating systems so that it can be recognized as a local disk. For example, an NAD device may treated as a local disk per se by a Windows 2006™ host so that all disk operations exercised by the host control a local disk, including formatting and partitioning, can be done to the NAD device.

This feature differentiates the present invention from other solutions, such as those provided by the NAS technology, which expand disk space through the intervention of a file system instead of directly adding individual disks at the device level of the host system. At the same time, since the NAD device is to be accessed through the network, the present invention redirects the disk I/O request to the network interface otherwise would be directed to the disk controller connected to the inside system bus in the case of using conventional local disks.

In other words, the present invention creates a virtual host bus adapter in purely software means by modifying a driver at the host so that the host recognizes the NAD device as if it is connected to the system bus through a physical host adapter although actually there is no physical host adapter connected to the bus. Since an NAD device is accessed as if it is a local device connected to the internal bus of a host, there is no need to use network addresses such as IP addresses for the host to communicate with the NAD device. Instead, data link frames containing storage commands are exchanged between the host and the NAD device.

FIGS. **20**A and **20**B shows a comparison of the hierarchy of the disk driver between the conventional disk system and the NAD system of the present invention. FIG. **20**A shows conventional disk driver layers in Windows 2000, which are organized in a hierarchy comprising a disk partition manager **601**, a disk class driver **602**, a port driver **603**, and a bus driver **604**.

In the Windows 2000 operating system, the generic term, 'bus', refers to a piece of hardware to which devices connect electronically. Not only does it include things like the PCI bus, but it also includes anything that can have multiple devices plugged into it such as a SCSI adapter, a parallel port, a serial port, a USB hub, and so on. One responsibility of the bus driver is to enumerate devices attached to the bus and to create physical device objects for each of them as necessary in Windows 2000. Therefore, the bus driver is a collection of software routines that contain the information about the specific bus and the functions that allocate system resources such as port addresses and IRQ numbers to the devices connected to the bus. It is the port driver that contains routines required to perform most of the actual disk I/O operations.

The major feature of the present invention is to replace the conventional bus driver and the port driver with a new bus driver and a new port driver so that the NAD devices can be recognized as the same as the local disks and the disk I/O operations can be performed to the NAD devices through the network port of the Windows 2000 host.

FIG. **20**B shows the driver layers of the present invention, which have an NAD port driver **613** and an NAD bus driver **614** replacing the corresponding conventional Windows 2000 driver layers of FIG. **20**A. The NAD bus driver **614** implements a virtual host bus adapter, through which disk I/O operations are to be done to and from a set of NAD devices. The NAD port driver **613** implements a set of routines required to perform actual disk I/O operations by redirecting the I/O requests to the NAD devices through the network port of a Windows 2000 host.

FIG. **21** shows an example of a network environment where NAD devices of the present invention are attached to multiple hosts. The example shows that both Host #**1** **621** and Host #**2** **622** run Windows 2000 connected to Network #**1** **623** and Network #**2** **624**. Host #**1** uses disk(**1**,**1**) **631** and disk(**1**,**3**) **633** through Network #**1**, disk (**2**,**1**) **636**, and disk(**2**,**2**) **637** through Network #**2** **625**. Similarly, Host #**2** uses disk(**1**,**2**) **632**, disk(**1**,**4**) **634** and disk(**1**,**5**) **635**.

Given the NAD bus driver and the NAD port driver, a Windows 2000 system creates device stacks as specified in Windows 2000 in order to be able to process I/O requests. Each device in Windows 2000 is expressed in terms of device objects organized in a stack structure. Device objects are data structures that the Windows 2000 system creates to help software manage hardware. Many of these data structures can exist for a single piece of physical hardware. The lowest-level device object in a stack is called a physical device object (PDO). Above a PDO in a device object stack is an object called a functional device object (FDO). There may be a collection of filter device objects below and above the FDO. The Plug and Play (PnP) Manager component of Windows 2000 constructs the stack of device objects at the command of device drivers. The various drivers that occupy the stack for a single piece of hardware perform different roles. The function driver manages the device, and the bus driver manages the connection between the device and the computer.

FIG. **22** shows an example of device stacks that may be created to implement the present invention, where all filter device objects are omitted for the simplicity. Shown on the left half is a layer of recursively enumerated SCSI devices on top of the PCI bus, which is typically the case when SCSI disks are connected to the host's PCI bus. In the first instance, a PnP Manager has a built-in driver for a virtual root bus that conceptually connects computer to all the hardware that can't electronically announce its presence including hardware bus such as PCI. The root bus driver **640** gets information about the PCI bus from the registry to create a PCI bus PDO **641**, a PDO for the PCI bus, where the registry was initialized by a Windows 2000 setup program.

Having created the PCI bus PDO **641**, the PnP Manager then loads functional drivers for the PCI bus, thus creating a PCI bus FDO **642**. The functional driver of the PCI bus can then enumerate its own hardware devices attached to the PCI bus, where the example system in FIG. **21** assumes to have a set of SCSI devices, to create a SCSI port PDO **643**. Once the SCSI port PDO **643** is created, the PnP Manager then loads drivers for SCSI port device, thus creating a SCSI port FDO **644**. Similarly, SCSI disk PDOs, such as **645** and **646**, are created for each of the individual SCSI disks on top of the SCSI port, and SCSI disk FDOs, such as **647** and **648**, are in turn created by loading the disk class driver.

Shown on the right half of FIG. **21** is the corresponding device stacks for the NAD devices that would be created by using the NAD bus driver and NAD port driver replacing the PCI bus driver and the SCSI port driver, respectively. On top of the root is a NAD BUS PDO **651**, the PDO of the NAD bus that is not conventional hardware bus such as PCI, but a bus required to fit in the Windows 2000 device stack in order to provide virtual bus for NAD devices. On top of the NAD bus PDO **651**, the PnP Manager creates a NAD bus FDO **652** by loading a NAD bus driver.

A set of NAD Port PDOs **653** and **654** for each of individual network interface cards (NICs) of the Windows 2000 host are then created recursively since one NAD port is implemented to correspond to one NIC of the host in the present invention. On top of each NAD port PDO such as **653** or **654**, each NAD port FDO such as **655** or **656** is created by loading a NAD port driver. It is the NAD port driver that performs the actual NAD disk I/O operations. The NAD port driver should handle the

NAD device I/O requests by redirecting the I/O requests and obtaining the I/O replies to and from the corresponding NAD devices through the specific NIC. The NAD port FDO such as **655** or **656** then creates individual NAD device PDOs such as **657**, **658**, **659** or **660** on top of the specific NAD port for individual NAD devices that can be accessed through the specific NAD port bound to a specific NIC.

FIG. **22** shows that for the example in FIG. **21**, two stacks of NAD port objects **653** and **654** are created because Host #**1** has two NICs. Host #**1** also has four NAD device PDOs **657** through **660**, two for each NAD port, because NAD devices, i.e., disk(**1**,**1**) **631** and disk(**1**,**3**) **633** and disk(**2**,**1**) **636** and disk(**2**,**2**) **637** are to be accessed through the NIC(**1**,**1**) and NIC(**1**,**2**) respectively. For each individual NAD device PDO such as **657**, **758**, **659** or **660**, the PnP Manager loads disk class driver to create a NAD device FDO such as **661**, **662**, **663** or **664**.

Note here that the only NAD bus driver and NAD port driver are to replace the conventional bus driver and SCSI port driver respectively in order to substitute the NAD devices for the conventional local disks. Disk class driver and other higher level drivers of Windows 2000 should remain intact without a single change in order for the Windows 2000 system to recognize the NAD device as same as a local disk.

In Windows 2000, each request for an operation affecting a device uses an I/O request packet (IRP). IRPs are normally sent to the topmost driver of a stack for the device and can percolate down the stack to the other drivers. At each level, the driver decides what to do with the IRP. Sometimes, the driver does nothing except passing the IRP down. The driver may completely handles the IRP without passing it down or process the IRP and pass it down. In the case of disk I/O, for example as shown in FIG. **20**B, an IRP for a file I/O sent to the file system driver is passed to a volume manager, a disk class partition manager, to a partition manager, and to disk class driver.

It is the disk class driver where a SCSI Request Block (SRB) is created to be included in the IRP as necessary. An SRB is a data structure specified in the Windows 2000 for SCSI device I/O. If the IRP is for the conventional local disk, the disk class driver passes the new IRP down to a SCSI port driver that completes actual disk I/O operation. If the IRP is for the NAD device connected to the network, the disk class driver passes the IRP down to NAD port driver that completes NAD device I/O through the network interface.

Without regard to the particular device type of the disk, local or NAD device, it is the feature of the Windows 2000 device stack as shown in FIGS. **20**A and **20**B that an IRP for a specific disk, local or network-attached, is directed eventually to the corresponding disk. This is because separate disk object stacks are created for each of the individual disks. FIG. **22** shows that separate SCSI disk FDO/PDOs and NAD device FDO/PDOs are bound to each of the individual local disks and NAD devices, respectively.

The present invention replaces the conventional disk bus driver and port driver with the new NAD bus and port drivers as shown in FIG. **20**B so that NAD devices would be recognized as local disks by the Windows 2000 system.

All of the Windows 2000 device drivers have functions to create and remove the FDO for each device and dispatch functions to handle IRP passed down from the above driver layer. The major and minor function numbers in the IRP specify which of the dispatch functions will be invoked.

The following is an explanation of the actual software modules implemented in the NAD bus driver and port driver of the present invention in order to implement the NAD system for Windows 2000.

NAD Bus Driver

The NAD bus driver is a set of software modules that implement a virtual host bus adapter to which NAD ports are to be attached, where the individual NICs of a host are realized as NAD ports. The functions of the NAD bus driver are basically the same as those of a conventional PCI bus driver in Windows 2000. The NAD bus driver performs the functions of finding out the number of the NICs installed in the host computer and enumerating those NICs to create an NAD port PDO for each of the existing NICs. It also performs the functions of creating, starting, stopping, and removing an NAD port. In the NAD system, an individual NIC is regarded as an independent NAD port so that NAD disk ports for NAD devices are created according to the number of independently operating network units. See the example configuration shown FIG. **22**.

The difference between the NAD bus driver and a conventional PCI bus driver is that the NAD bus driver is for NAD devices that are physically separated from the system bus of the host but are connected through network ports. Unlike a conventional Windows 2000 system that detects plug-in of a device to or removal of a device from the hardware bus through a hardware interrupt, the NAD bus driver is implemented by creating a kernel thread to install and remove an NAD port on the NAD bus. The kernel thread created by the NAD bus driver starts to work when an IRP with IRP_MJ_PNP as its major function number and IRP_MN_START_DEVICE as its minor function number is sent to the NAD bus FDO from the PnP Manager. The thread terminates when the NAD bus FDO is removed. The thread periodically detects existence of NICs. If a new NIC is detected, the thread creates a new NAD port PDO for the NIC and includes the newly created NAD port PDO into its own list of NAD port PDOs. The thread then invokes the PnP Manager to have the NAD port PDO recognized by the system. Removal of an NIC is also detected by the thread since the thread can detect the absence of the NIC of which NAD port PDO previously created would be found in the above mentioned list without the corresponding NIC. If an NIC is found to have been removed from the host, the thread removes the corresponding NAD port PDO from its list and invokes the PnP Manager to remove the NAD port from the Windows 2000 system.

The software routines implemented in the NAD bus driver can be classified into five categories. The following tables list some of the routines implemented in the NAD bus driver with brief explanations.

TABLE 4

Basic functions	
DriverEntry ( )	executed when the driver is initially loaded registers dispatch routines of the NAD bus driver initializes the variables used by the driver
NADBusUnload ( )	recovers resources occupied by the driver when the driver is unloaded
NADBusAddDevice ( )	creates NAD bus FDO initializes the value of the NAD bus FDO

TABLE 5

Dispatch functions	
NADBusCreate ( )	processes the 'IRP__MJ__CREATE' IRP
NADBusClose ( )	processes the 'IRP__MJ__CLOSE' IRP

TABLE 5-continued

Dispatch functions	
NADBusPnp ( )	processes the 'IRP__MJ__PNP' IRP determines whether the IRP passed is to NAD bus FDO or to NAD port PDO, and invokes NADFDOPnP( ) or NADPDOPnP( ) accordingly
NADBusPower ( )	processes 'IRP__MJ__POWER' IRP

TABLE 6

NAD bus FDO related functions		
NADBusFDOPnp( )	invoked when IRP__MJ__PNP is sent to NAD bus FDO processes various minor functions according to the minor function number sent together	
	IRP__MN__START__DEVICE	transfer NAD bus FDO to 'started' state - invokes NADBusStartFDO( )
	IRP__MN__QUERY__STOP__DEVICE	invoked to query if NAD bus FDO can be stopped transfer NAD bus FDO to 'stop pending' state
	IRP__MN__CANCEL__STOP__DEVICE	invoked to cancel IRP__MN__QUERY__STOP__DEVICE
	IRP__MN__STOP__DEVICE	stops NAD bus FDO - transfers NAD bus FDO to 'stopped' blocks NADBusHW( ) thread
	IRP__MN__QUERY__REMOVE__DEVICE	invoked to query if NAD bus FDO can be removed from the system
	IRP__MN__CANCEL__REMOVE__DEVICE	invoked to cancel IRP__MN__QUERY REMOVE__ DEVICE
	IRP__MN__SURPRISE REMOVAL	invoked when NAD bus FDO is removed abnormally
	IRP__MN__REMOVE__DEVICE	invoked when NAD bus FDO is removed normally
	IRP__MN__QUERY__DEVICE__RELATIONS	passes list of NAD port PDO to PnP manager
NADBusStartFdo( )	allocates resources to NAB bus FDO	
NADBusRemoveFdo( )	recovers resources occupied by NAD bus FDO removes the NADBusHW( ) thread	
NADBusGetDeviceCapabilities( )	passes DeviceCapability dtructure to PnP manager	

TABLE 7

NAD port PDO related functions	
NADPortPDOPnp( )	processes minor functions related to PnP invoked when IRP__MJ__PNP is sent to NAD port PDO Minor functions: IRP__MN__START__DEVICE IRP__MN__QUERY__STOP__DEVICE IRP__MN__CANCEL__STOP__DEVICE IRP__MN__STOP__DEVICE IRP__MN__QUERY__REMOVE__DEVICE IRP__MN__CANCEL__REMOVE__DEVICE IRP__MN__SURPRISE__REMOVAL IRP__MN__REMOVE__DEVICE IRP__MN__QUERY__CAPABILITIES IRP__MN__QUERY__ID IRP__MN__QUERY__DEVICE__RELATIONS IRP__MN__QUERY__DEVICE__TEXT IRP__MN__QUERY__RESOURCES__REQUIREMENTS IRP__MN__QUERY__RESOURCE
NADPortPDOQueryDeviceCaps( )	returns DEVICE__CAPABILITIES data structure of NAD port
NADPortPDOQueryDeviceId( )	returns device ID, instance ID, hardware ID of NAD port
NADPortPDOQueryDeviceText( )	returns location and description of NAD port
NADPortPDOQueryDeviceRelations( )	returns target device relation value
NADPortInitializePdo( )	initialize NAD port PDO value invoked when NAD port attached to NAD bus is detected
NADPortDestroyPdo( )	removes NAD port PDO and recovers resources

TABLE 8

Function to detect NAD port	
NADBusHW( )	routine for the kernel thread to detect NAD ports attached to NAD bus - periodically detects the existence of NICs if a new NIC is detected, creates NAD port PDO and invokes NADPortInitializePdo( ) if a NIC is detected to have been removed, removes NAD port PDO by invoking NADPortDestroyPdo( )

NAD Port Driver

A port driver is a lower-level driver that responds to a system-defined device control request or a driver-defined device I/O control request from a corresponding class driver.

The NAD port driver is capable of basic functions to initialize the driver and create an NAD port FDO and dispatch functions to process IRP passed down from the disk class driver layer. The IRP passed down from the disk class driver may contain a SCSI request block (SRB), which specifies the actual I/O command to be performed onto the SCSI device.

Tables 9 and 10 list the basic functions and some of the dispatch functions, of which roles are basically the same as those of the NAD bus driver described earlier, are presented with brief explanations.

TABLE 9

Basic functions	
DriverEntry( )	initializes driver registers driver functions
NADPortAddDevice( )	invoked by PnP manager to create NAD port FDO
NADPortDriverUnload( )	invoked when to remove driver recovers resources

TABLE 10

Dispatch functions for initialization, creation, and removal of the NAD port	
NADPortCreateClose( )	processes IRP_MJ_CREATE and IRP_MJ_CLOSE IRP
NADPortCleanup( )	processes IRP_MJ_CLEANUP IRP recovers resources

TABLE 10-continued

Dispatch functions for initialization, creation, and removal of the NAD port	
NADPortPnp( )	processes IRP_MJ_PNPIRP
NADPortPower( )	processes IRP_MJ_POWER IRP

In Windows 2000, a device I/O control command is included in an IRP as a device I/O control number, and the device I/O control functions are implemented in the port driver to handle the corresponding device I/O control numbers.

Besides the regular device I/O control functions in Windows 2000, additional device I/O control functions are implemented in the NAD port driver so that the NAD can be added or removed dynamically without stopping the Windows 2000 system. With conventional local disks, addition or removal of the local disk can be directly detected by the Windows 2000 system at the time of the system booting because the local disks are physically connected to the physical hardware bus. Therefore, the creation of a disk PDO for a local disk is basically initiated from the hardware interrupt at the time of the system booting. So the conventional port driver does not have to have functions that initiate addition or removal of the PDO of a disk device in the middle of the system operation.

However, in an NAD system, addition and removal of an NAD device can occur while the Windows 2000 system is running. Therefore, there should be a mechanism that can create/remove a disk PDO for the newly attached/removed disk.

The device I/O control functions implemented in the present invention handle such dynamic addition and removal of the NAD as follows. If a device control IRP that tells a new NAD hardware device is hooked up to the network is passed to the NAD port FDO, the NAD port FDO creates an NAD device PDO for the new NAD thus letting the system recognize the disk. For the removal of the NAD device, device control IRP to remove the disk is sent to and processed by NAD port FDO similarly.

The dispatch functions that handle device I/O control IRPs are summarized in the following table. Note that the device I/O control functions, NASPortFdoDeviceControl( ), NADPortPluInDevice( ), and NADPortUnpluDevice( ) are the functions particular to the present invention for the purpose of dynamic addition and removal of the NAD.

TABLE 11

NAD port device control dispatch functions	
NADPortDeviceControl( )	invoked when I/O control IRP is passed processes IRP_MJ_DEVICE_CONTROL IRP invokes NADPortFdoDeviceControl( ) for FDO control invokes NADPortPdoDeviceControl( ) for PDO control
NADPortFdoDeviceControl( )	registers new NAD device or removes an NAD device processes I/O control functions IOCTL_NADPORT_PLUGIN_HARDWARE Invokes NADPortPlugInDevice( ) to register new NAD hardware IOCTL_NADPORT_UNPLUG_HARDWARE invokes NADPortUnplugDevice( ) to remove an NAD hardware
NADPortPdoDeviceControl( )	processes I/O controls for PDO invokes 110 control functions according to the device I/O control function numbers in the IRP IOCTL_STORAGE_QUERY_PROPERTY queries NAD device property IOCTL_GET_DISK_DRIVE_GEOMETRY returns DISK_GEOMETRY data structure containing geometry information of the NAD device

TABLE 11-continued

NAD port device control dispatch functions
IOCTL_GET_SCSI_ADDRESS NAD device does not use SCSI address, so sets the values of PathID and TargetID 0s and returns enumeration number of NAD device to LUN in CDB

FIG. **23**A shows the flow of IRP, SRB, and CDB where the IRP is to a SCSI disk connected to a conventional hardware bus such as PCI bus in a Windows 2000 system. A disk class driver **681** passes down to a SCSI port driver **682** and a SCSI bus driver **683** an IRP that may contain an SRB. The SRB is a data structure that contains information about the requested I/O and a command descriptor block (CDB) containing a SCSI-2 standard command. Receiving the IRP with SRB from the disk class driver, the SCSI port driver **682** and the SCSI bus driver **683** deliver the CDB extracted from the SRB to the SCSI host adapter **684** to complete an actual device I/O to a SCSI disk **685**.

In a conventional local disk, disk I/O commands are delivered to a disk controller at the host adapter using the SRB data structure. But, in the NAD system of the present invention, disk input/output commands are delivered to the NIC of the host.

FIG. **23**B shows the flow of IRP, SRB, and CDB (or some other types of I/O commands) in the NAD system. A disk class driver **691** passes down an IRP with an SRB to a NAD port driver **692** and a NAD bus driver **693**, which then deliver the CDB extracted from the SRB to NIC **694** to complete an actual device I/O to a NAD device **695** through a network **696**.

In the present invention, the NAD system supports various types of disks including SCSI and IDE. If the NAD device is composed of SCSI disks only, the CDB is delivered as is to the host NIC so that the network-attached SCSI disks can perform the requested disk I/O.

If the NAD device, however, is composed of disk devices of other type than SCSI such as IDE, the CDB must be translated into the commands that can be processed by the specific devices. The translation of the CDB, in such a case, can be done either at the NAD port driver or at the NAD device. If the translation is to be done at the NAD, the Windows 2000 host simply delivers a CDB to the host NIC as if it delivers a CDB to a SCSI disk. If the translation is to be done at the NAD port driver, the NAD port driver functions must translate the CDBs into a set of disk I/O commands appropriate to the specific hardware disk types.

The NAD system of the present invention supports both cases, and the type of the commands, i.e., CDB or hardware-specific commands, is determined at the time of the installation of the specific NAD. Some of the dispatch functions that process SRB with mandatory CDB operation codes are given in table 12 to show how the NAD port driver functions are implemented to handle the SRB and CDB in the present invention. Such SRB processing functions are required if the NAD port driver has to translate the CDB into a set of hardware specific I/O commands.

TABLE 12

NADPortInternalDeviceControl( ) - executes SrbFunctionExecuteScsi( ) when SRB_FUNCTION_EXECUTE_SCSI is passed as the SRB function value		
SrbFunctionExecuteScsi( )	processes CDB invokes CDB processing functions according to the CDB operation codes given below	
	SISCOP_TEST_UNIT_READY	tests if an NAD device is accessible
	SCSIOP_MODE_SENSE	returns configuration of NAD device
	SCSIOP_READ	reads a block from NAD
	SCSIOP_WRITE	write a block to NAD
	SCSIOP_MODE_SELECT	sets parameter to NAD
	SCSIOP_READ_CAPACITY	returns size of the next block or address of the last block
	SCSIOP_REASSIGN_BLOCKS	relocates block
	SCSCIOP_RESERVE/SCSIOP_RELEASE	changes status information
	SCSIOP_START_UNIT	starts NAD
	SCSIOP_STOP_UNIT	stops NAD
	SCSIOP_VERIFY	verifies data stored in NAD

Communication Between the Host and the NAD

Disk I/O commands in the NAD system are delivered to the host NIC instead of the local disk host adapter because the I/O should be done over the network rather than over the bus. Windows 2000 provides a Network Driver Interface Specification (NDIS), a set of specifications defined to specify network interface drivers.

FIG. **24** shows a NDIS driver layer defined in Windows 2000. It consists of a NDIS protocol driver **701** for specifying a high-level protocol to be used, a NDIS intermediate driver **702**, an NDIS miniport **703** for managing hardware specifics, and a network interface card (NIC) **704**.

In the present invention, all the NAD port driver functions that deliver I/O commands to the NAD devices are implemented to deliver the commands to a NDIS (network driver interface specification) protocol driver layer through which the commands are delivered to the NAD devices over the network.

Upon receiving from the disk class driver the IRP containing an SRB or an I/O control command for specific disk I/O operation, the NAD port driver passes down a new IRP containing the corresponding CDB to the protocol driver. Then the protocol driver sends the CDB, which is the SCSI-2 standard I/O command, to the NAD device and, in turn, receives and handles the reply from the NAD device. Note here that if the host computer has to send some hardware specific I/O commands other than CDB as is pointed out in FIG. **23**B, the NAD port driver passes down an IRP containing the hardware specific commands instead of the CDB to the NDIS protocol driver.

The NDIS provides transport-independence for network vendors because all drivers that require communication over the network calls the NDIS interface to access the network, thus providing a ready solution for the communication between the host computer and the NAD devices in the present invention.

The actual protocol implemented in the protocol driver of the NDIS may adopt a standard protocol or a non-standard protocol. Since a standard protocol such as IP (Internet Protocol) involves an overhead, a non-standard protocol may be preferred in terms of performance and security. The present invention follows the NDIS specification of the Windows 2000 network system to implement a proprietary communication protocol into the NDIS protocol driver in order to provide a communication protocol between a Windows 2000 host and NAD devices to reliably handle the NAD I/O commands.

NAD Device

The technical constitution of the NAD device running under the Windows family of operating systems is the same as that of the NAD device running under the UNIX family of operating systems shown in FIG. **14**.

Advantages of the NAD System Over NAS and SAN

Either running the UNIX or Windows family of operating systems, the NAD system of the present invention has numerous advantages over the NAS system and the SAN system. Unlike the NAS system that provides file storage service by way of an additional file server, the NAD device is attached to a host computer as if it is a local disk connected to the system bus of the host. Unlike the SAN system, the NAD device of the present invention is simply plugged into a network port without requiring any additional special switch or network equipment. Therefore, the NAD system provides better user convenience, system flexibility, scalability, economy, and performance.

All the disk-related operations, including formatting, partitioning, sharing, and mounting, can be done to NAD devices just as they can be done to a local disk. Since NAD devices are directly available to the host as local disks, the NAD system provides better manageability and user convenience. In the NAS system, addition, deletion, or any change to the disk configuration should be consulted to the NAS operating system through human or software intervention. In the NAD system, addition or deletion of an NAD device is instantly achieved by plugging or unplugging the NAD device to and from a network port. The NAD system even provides a superior user convenience in installing and uninstalling the disks, eliminating the need of opening and closing the case of the host computer.

The NAD system provides almost unlimited scalability to the disk capacity. The number of NAD devices that can be attached to the network is virtually unlimited, whereas the number of disks available through the NAS system is severely limited because of an economical reason and the inconvenience involved in the management of the multitudes of NAS servers.

The NAD system is intrinsically more economical than the NAS or SAN system because each NAD device does not employ file server software and other additional special hardware equipment.

Media Changeable NAD System

An NAD system of the present invention can be alternatively implemented as a media changeable storage device. A media changeable storage device is a special storage device that is physically separated two parts, one being the media containing the data and the other being the driver performing an I/O operation to the media. Floppy disk drivers, CD-ROM drivers are examples of media changeable storage devices. Whether a media is installed or not, a media changeable storage device can be registered to a host computer so that a media such as a diskette can be inserted into a driver dynamically.

Since NAD devices can be plugged in or removed from a network port dynamically, a virtual driver that uses NAD as a media can be implemented in the form of a media changeable storage device. Windows 2000 provides the changer class driver model to implement a media changeable storage device. In order to implement a media changeable NAD system, a class driver for the NAD system is implemented according to the model of the changer class driver of Windows 2000. The two lower level drivers, i.e. the NAD port driver and the NAD bus driver, are used to implement such media changeable NAD system.

Alternative Embodiment Using Converter and Counter-Converter

Instead of using a network interface card (NIC) and new virtual host bus adapter, the network attached disk of the present invention may be implemented by providing in the host side a protocol converter that converts storage commands into data link frames containing the storage commands so that the frames can be sent through a network, and by providing in the device side a counter-converter that converts the data link frames containing the storage commands received through the network into the storage commands.

Since a converter is a specialized network interface, the converter encapsulates the I/O commands and data to data link frame so as transmit them to an I/O device through a network without the overhead of processing communication protocols in general.

Tape System, CD Juke Box

The kinds of storage devices that can be directly connected to a network using the interface of the present system are not limited to disk systems. Tape systems and CD drives use IDE or SCSI interface, the same bus interface as disk systems. For example, the present invention may be used to connect multiple CD drives directly to a network, enabling a cost-effective implementation of a CD-Juke box.

While the invention has been described with reference to preferred embodiments, it is not intended to be limited to those embodiments. It will be appreciated by those of ordinary skilled in the art that many modifications can be made to the structure and form of the described embodiments without departing from the spirit and scope of this invention.

The invention claimed is:

1. A method of accessing a network coupled device through a network carrying general-purpose network traffic using a certain network protocol from a host having an internal host system bus and running an operating system, the method comprising:

operating a virtual host bus adapter at a device driver level for controlling the device through the network via a network interface, the virtual host bus adapter including a device driver enumerating devices that are available over the network, not directly attached to the host internal system bus, so that the host recognizes the device as a host local device, the virtual host bus adapter controlling the device in a way indistinguishable from the way it is controlled as a physical host bus adapter device controlling the same type of device so that the host recognizes the device as if it is a local device connected directly to the system bus of the host;

encapsulating device level commands and optional data for controlling the device into outgoing data link frames for communication over the network;

communicating the outgoing data link frames on the network;

receiving the outgoing data link frames and extracting the device level commands and optional data;

communicating the extracted device level commands and optional data to the device and operating the device in accordance with the extracted device level commands;

generating an incoming command and optional data and encapsulating it into an incoming data link frame for communicating to the host over the network;

sending the incoming data link frame to the host; and

extracting the device level command and optional data from the host received data link frame and processing the device level command in accordance with predetermined rules.

2. The method of claim **1**, wherein the device level commands contained in and extracted from the data link frames are the same device level commands that are used to access the same type of device when such same type of device is connected to the host over the internal host system bus.

3. The method of claim **1**, wherein the operating system is a member of the UNIX family of operating systems.

4. The method of claim **1**, wherein the operating system is a member of the Windows family of operating systems.

5. The method of claim **4**, wherein the virtual host adapter is configured to operate according to the Network Driver Interface Specification.

6. The method of claim **1**, wherein data link frames are compatible with TCP/IP.

7. The method of claim **1**, wherein the network coupled device is a disk.

8. The method of claim **1**, wherein the network coupled device is a tape device.

9. The method of claim **1**, wherein the network coupled device is a compact disk drive.

10. The method of claim **1**, wherein the network coupled device is a digital versatile disk device.

11. The method of claim **1**, wherein the network coupled device is a memory device.

* * * * *

EXHIBIT K

US007457880B1

(12) **United States Patent**
Kim

(10) **Patent No.:** **US 7,457,880 B1**
(45) **Date of Patent:** **Nov. 25, 2008**

(54) **SYSTEM USING A SINGLE HOST TO RECEIVE AND REDIRECT ALL FILE ACCESS COMMANDS FOR SHARED DATA STORAGE DEVICE FROM OTHER HOSTS ON A NETWORK**

(75) Inventor: **Han-gyoo Kim**, Irvine, CA (US)

(73) Assignee: **Ximeta Technology, Inc.** (VG)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 200 days.

(21) Appl. No.: **10/951,474**

(22) Filed: **Sep. 27, 2004**

Related U.S. Application Data

(60) Provisional application No. 60/590,722, filed on Jul. 22, 2004, provisional application No. 60/581,691, filed on Jun. 21, 2004, provisional application No. 60/506,829, filed on Sep. 26, 2003.

(51) **Int. Cl.**
G06F 15/16 (2006.01)

(52) **U.S. Cl.** **709/229**; 709/216; 709/225

(58) **Field of Classification Search** 709/225, 709/229, 224, 216; 707/3, 8, 100; 726/16; 718/104, 100; 710/200
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,329,619 A 7/1994 Page et al.
5,426,427 A 6/1995 Chinnock et al.
5,463,772 A 10/1995 Thompson et al.
5,513,314 A 4/1996 Kandasamy et al.
5,524,247 A * 6/1996 Mizuno 710/200
5,566,331 A 10/1996 Irwin et al.
5,721,818 A 2/1998 Hanif et al.
5,774,660 A 6/1998 Brendel et al.
5,781,550 A 7/1998 Templin et al.
5,812,930 A 9/1998 Zavrel
5,838,916 A 11/1998 Domenikos et al.
5,889,942 A 3/1999 Orenshteyn

(Continued)

FOREIGN PATENT DOCUMENTS

DE 19610840 9/1997

(Continued)

OTHER PUBLICATIONS

Japanese Office Action, App. No. 555298/2002, Jan. 9, 2007.

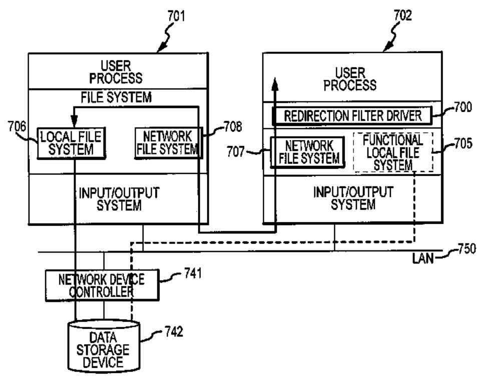
(Continued)

Primary Examiner—Le Luu
(74) *Attorney, Agent, or Firm*—Dorsey & Whitney LLP

(57) **ABSTRACT**

Access by multiple hosts, such as computers, to a data storage device by way of a network while maintaining data integrity. In one embodiment, a method for accessing the storage device includes acquiring a resource "lock" that provides exclusive access to one of the hosts at a time. In another embodiment, the file systems of a first and second host provide file system attributes stored in a storage device to provide mutually exclusive access for each host to free blocks of the device. In another embodiment, a networked system contains a first host having exclusive direct access to a storage device over a digital network. A second host requiring access to the storage device communicates with the first host by way of the digital network. File access requests generated by the second host are transferred by a redirection filter driver within the second host to the first host.

29 Claims, 14 Drawing Sheets



U.S. PATENT DOCUMENTS

5,987,523	A		11/1999	Hind et al.	
5,987,627	A		11/1999	Rawlings, III	
5,999,808	A		12/1999	Ladue	
6,047,307	A	*	4/2000	Radko	718/100
6,085,234	A		7/2000	Pitts et al.	
6,128,644	A		10/2000	Nozaki	
6,128,690	A		10/2000	Purcell et al.	
6,167,490	A		12/2000	Levy et al.	
6,175,869	B1		1/2001	Ahuja et al.	
6,216,202	B1		4/2001	E'Errico	
6,314,465	B1		11/2001	Paul et al.	
6,317,775	B1		11/2001	Colie et al.	
6,327,594	B1		12/2001	Van Huben et al.	
6,345,300	B1		2/2002	Bakashi et al.	
6,347,095	B1		2/2002	Tang et al.	
6,356,915	B1		3/2002	Chrchetkine et al.	
6,360,265	B1		3/2002	Falck et al.	
6,366,988	B1		4/2002	Skiba et al.	
6,389,432	B1		5/2002	Pothapragada et al.	
6,393,569	B1		5/2002	Orenshteyn	
6,404,766	B1		6/2002	Kitai et al.	
6,421,753	B1		7/2002	Hoese et al.	
6,449,647	B1		9/2002	Colby et al.	
6,470,389	B1		10/2002	Chung et al.	
6,510,164	B1		1/2003	Ramaswamy et al.	
6,518,965	B2		2/2003	Dye et al.	
6,523,066	B1	*	2/2003	Montroy et al.	709/229
6,529,996	B1		3/2003	Nguyen et al.	
6,539,446	B1	*	3/2003	Chan	710/200
6,578,111	B1		6/2003	Damron et al.	
6,594,677	B2		7/2003	Davis et al.	
6,598,068	B1	*	7/2003	Clark	718/104
6,609,167	B1		8/2003	Bastiani et al.	
6,647,016	B1		11/2003	Isoda et al.	
6,732,104	B1		5/2004	Weber	
6,760,783	B1		7/2004	Berry	
6,807,581	B1		10/2004	Starr et al.	
6,823,458	B1	*	11/2004	Lee et al.	726/16
6,834,326	B1		12/2004	Wang et al.	
6,894,981	B1		5/2005	Coile et al.	
6,941,576	B2		9/2005	Amit	
7,010,303	B2		3/2006	Lewis et al.	
7,069,312	B2		6/2006	Kostic et al.	
7,069,350	B2		6/2006	Fujita et al.	
7,076,690	B1		7/2006	Todd et al.	
7,251,704	B2		7/2007	Solomon et al.	
7,277,955	B2		10/2007	Elliott	
7,376,133	B2		5/2008	Gettala et al.	
7,383,229	B2		6/2008	Jacoby	
2002/0069245	A1		6/2002	Kim	
2003/0014569	A1		1/2003	Kim	
2003/0018403	A1		1/2003	Braun et al.	
2003/0028614	A1		2/2003	Jeon	
2003/0172149	A1	*	9/2003	Edsall et al.	709/224
2003/0225834	A1		12/2003	Lee et al.	
2004/0068563	A1	*	4/2004	Ahuja et al.	709/225
2004/0111422	A1	*	6/2004	Devarakonda et al.	707/100
2004/0117813	A1		6/2004	Karaoguz et al.	
2004/0220933	A1	*	11/2004	Walker	707/8
2004/0267706	A1	*	12/2004	Springer et al.	707/3
2005/0042591	A1		2/2005	Bloom et al.	
2005/0110768	A1		5/2005	Marriott et al.	
2005/0149682	A1		7/2005	Kim	
2005/0193017	A1		9/2005	Kim	
2005/0193189	A1		9/2005	Kim	
2006/0004935	A1		1/2006	Seto et al.	
2006/0010287	A1		1/2006	Kim	
2006/0045130	A1		3/2006	Kim	
2006/0067356	A1		3/2006	Kim	
2006/0069884	A1		3/2006	Kim	
2006/0155805	A1		7/2006	Kim	
2007/0008988	A1		1/2007	Kim	

FOREIGN PATENT DOCUMENTS

DE	10038142	2/2001
JP	10-113469	5/1998
JP	10-271562	10/1998
JP	11007404	1/1999
JP	11-114224	4/1999
KR	10-2000-72493	12/2000
KR	10-2001-0088528	9/2001
WO	WO99/03297	7/1999
WO	WO00/29529	5/2000

OTHER PUBLICATIONS

Japanese Office Action, App. No. 513300/2003, Feb. 6, 2007.
Blunden et al.,"Storage Network Virtualization: What's it all about?", ibm.com/redbooks, XP-002286341, pp. 1-110, Dec. 2000.
Klein, Yaron, "Storage Virtualization with iSCSI Protocol", Internet Draft, XP-015030964, pp. 1-15, Nov. 2, 2000.
Schulz, Greg, "SAN and NAS; Complementary Technologies", http://www.mti.com/white_papers/WP20002.pdf, XP-002201566, pp. 1-7, May 1, 2000.

* cited by examiner

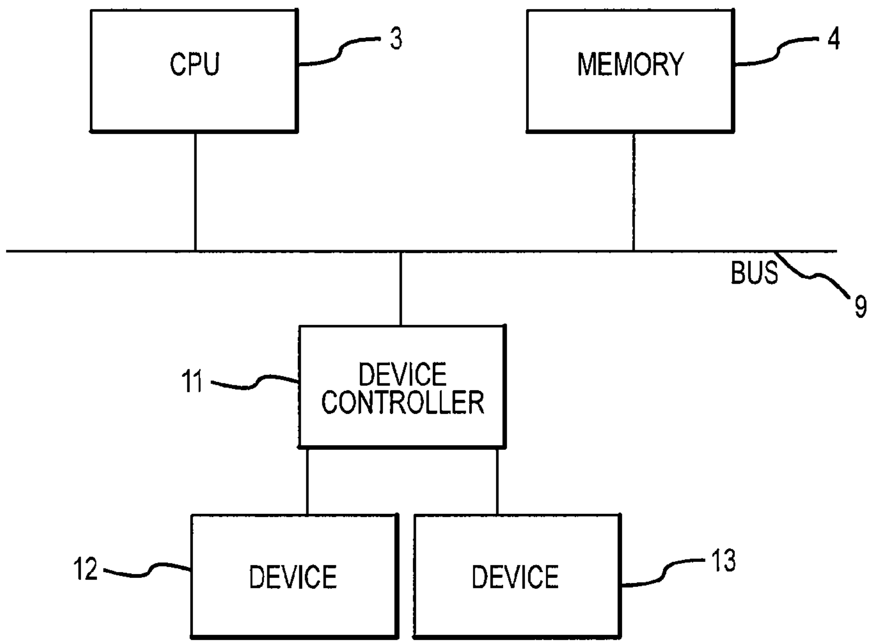


FIG.1
(PRIOR ART)

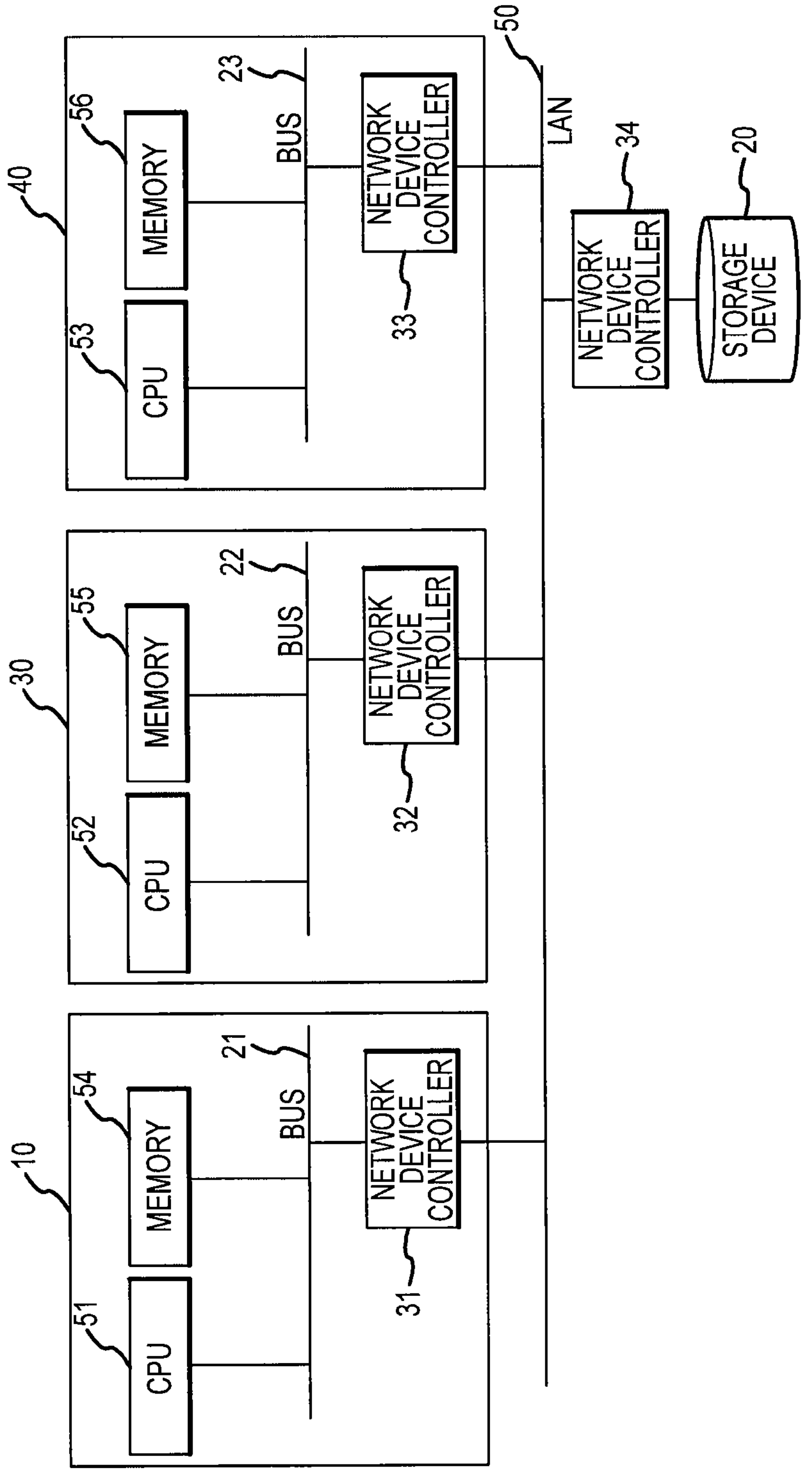

10
30
40
50
CPU
MEMORY
BUS
NETWORK DEVICE CONTROLLER
LAN
STORAGE DEVICE
51
52
53
54
55
56
21
22
23
31
32
33
34
20

FIG.2

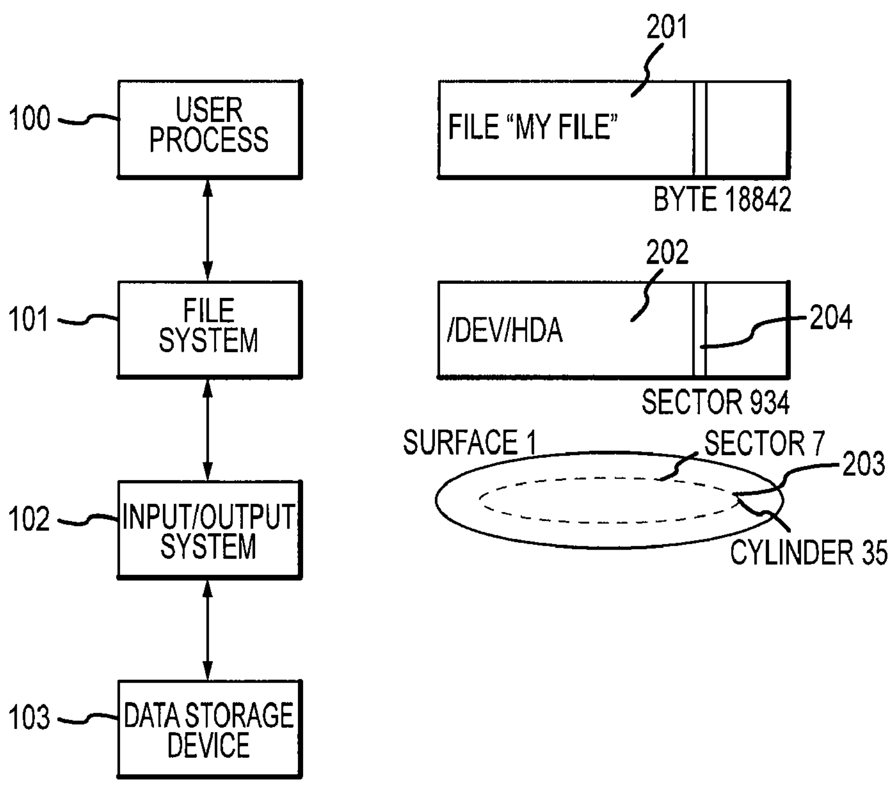

100
USER PROCESS
101
FILE SYSTEM
102
INPUT/OUTPUT SYSTEM
103
DATA STORAGE DEVICE
201
FILE "MY FILE"
BYTE 18842
202
/DEV/HDA
204
SECTOR 934
SURFACE 1
SECTOR 7
203
CYLINDER 35

FIG.3

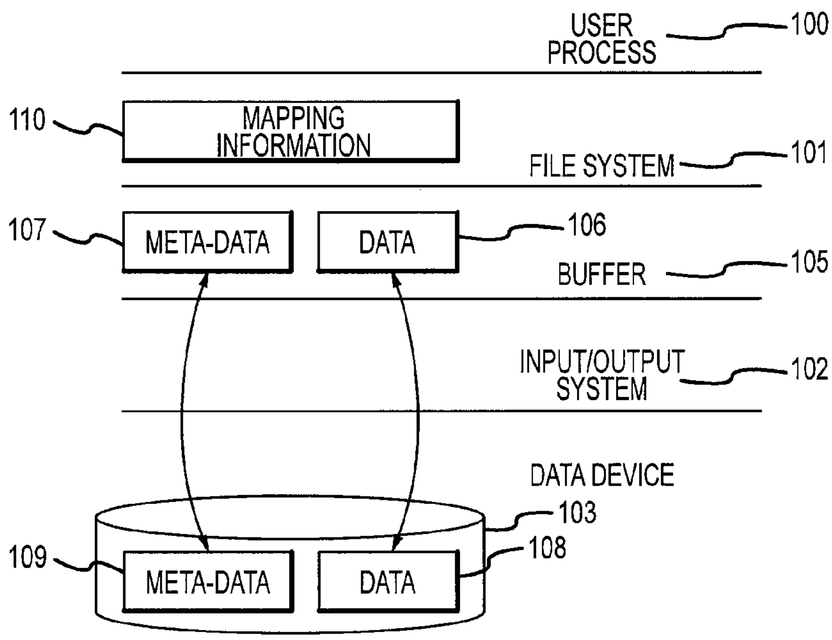

USER PROCESS
100
110
MAPPING INFORMATION
FILE SYSTEM
101
107
META-DATA
DATA
106
BUFFER
105
INPUT/OUTPUT SYSTEM
102
DATA DEVICE
103
108
109
META-DATA
DATA

FIG.4

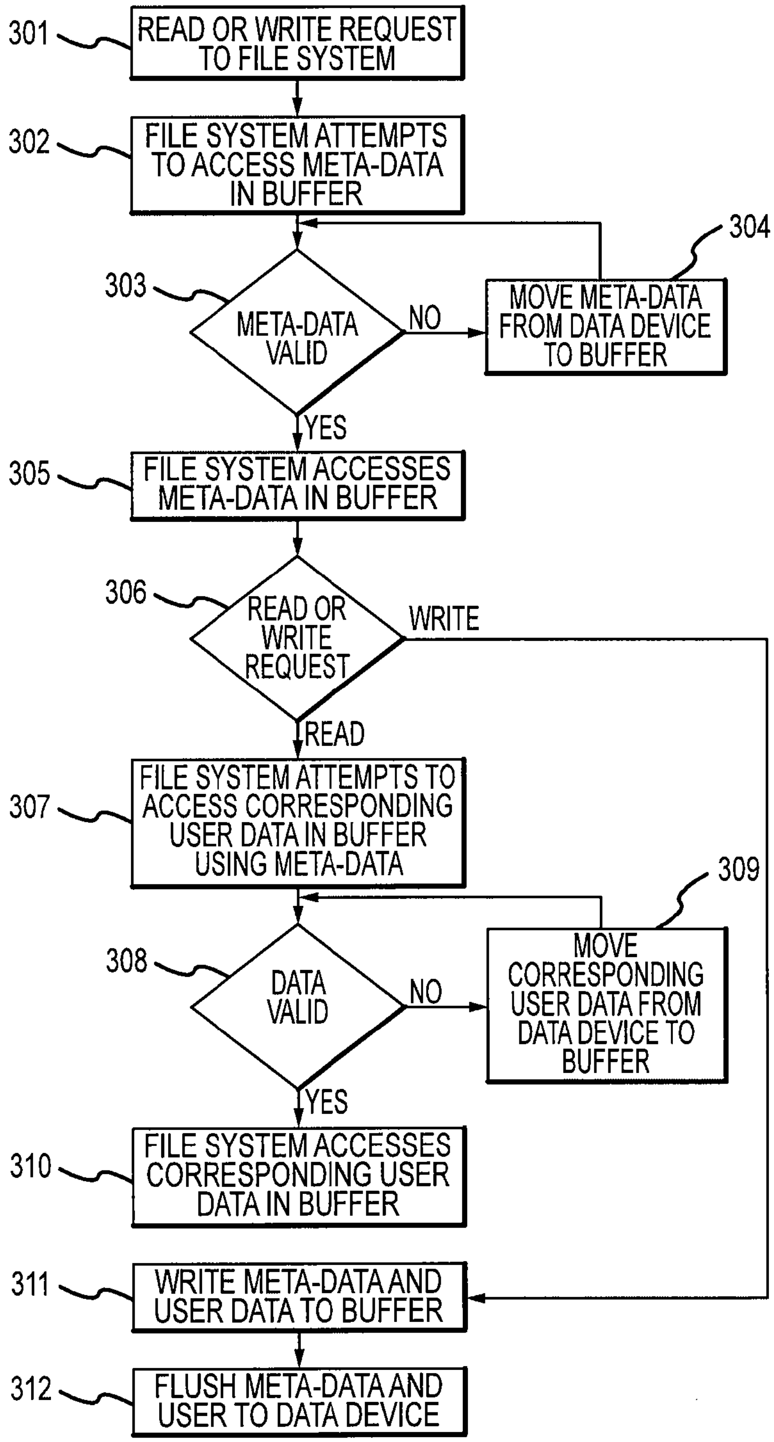


FIG.5
(PRIOR ART)

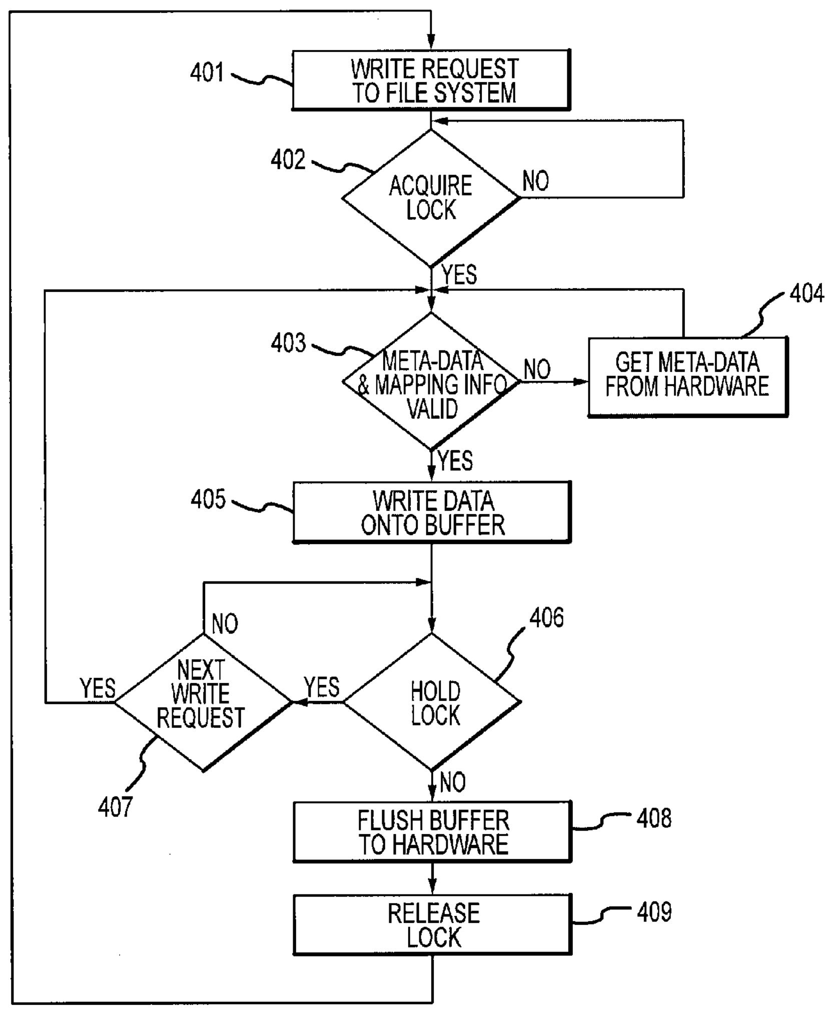
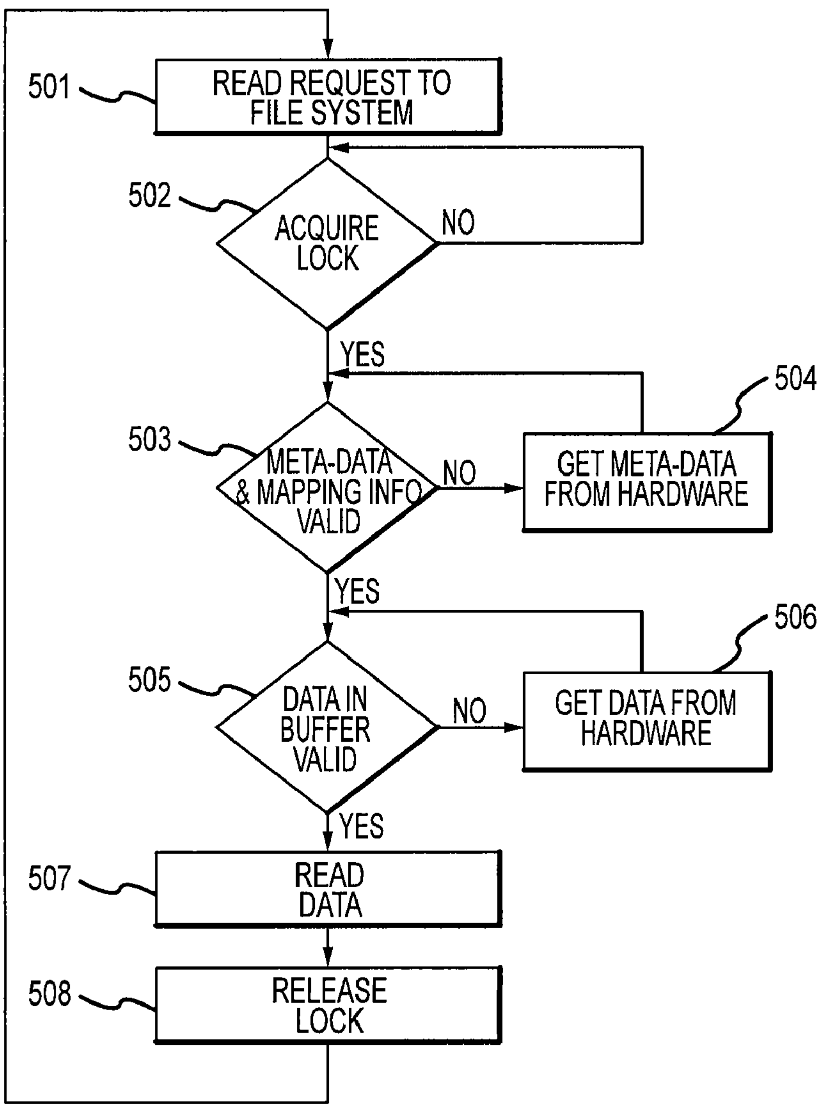

401
WRITE REQUEST TO FILE SYSTEM
402
ACQUIRE LOCK
NO
YES
403
META-DATA & MAPPING INFO VALID
NO
404
GET META-DATA FROM HARDWARE
YES
405
WRITE DATA ONTO BUFFER
NO
406
NEXT WRITE REQUEST
YES
YES
HOLD LOCK
407
NO
FLUSH BUFFER TO HARDWARE
408
RELEASE LOCK
409

FIG.6



501
READ REQUEST TO FILE SYSTEM
502
ACQUIRE LOCK
NO
YES
503
META-DATA & MAPPING INFO VALID
NO
504
GET META-DATA FROM HARDWARE
YES
505
DATA IN BUFFER VALID
NO
506
GET DATA FROM HARDWARE
YES
507
READ DATA
508
RELEASE LOCK

FIG.7

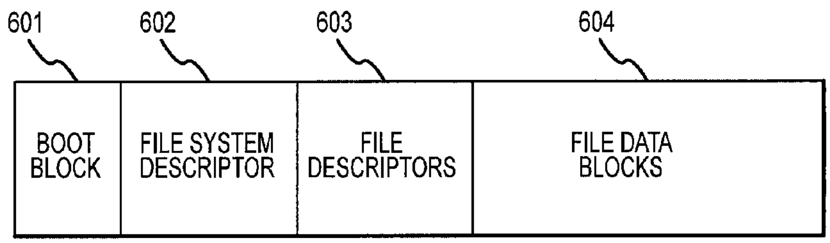

601
602
603
604
BOOT BLOCK
FILE SYSTEM DESCRIPTOR
FILE DESCRIPTORS
FILE DATA BLOCKS

FIG.8

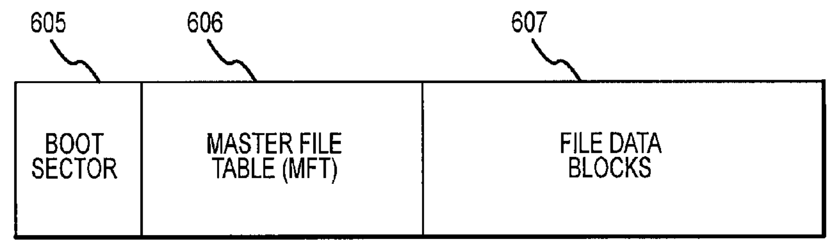

605
606
607
BOOT SECTOR
MASTER FILE TABLE (MFT)
FILE DATA BLOCKS

FIG.9

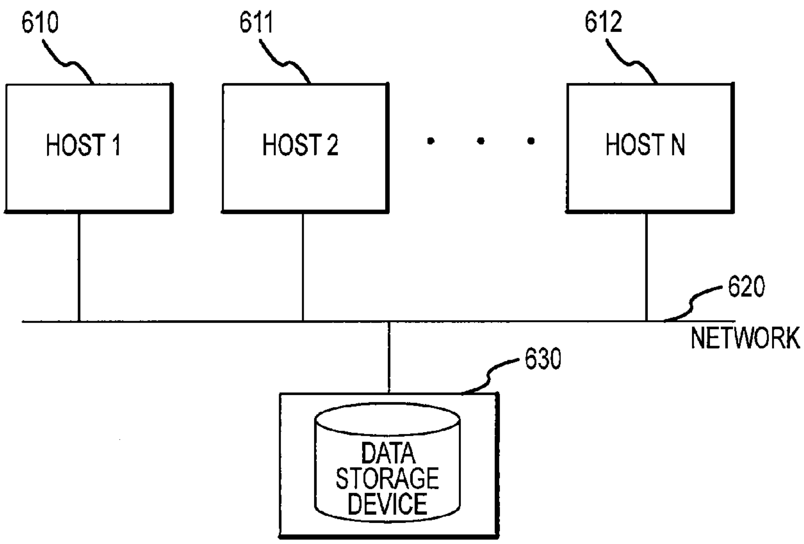

610
611
612
HOST 1
HOST 2
HOST N
620
NETWORK
630
DATA STORAGE DEVICE

FIG.10

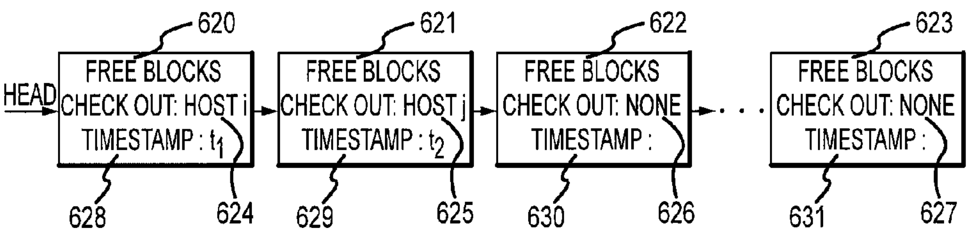

620
621
622
623
HEAD
FREE BLOCKS
CHECK OUT: HOST i
TIMESTAMP : t1
FREE BLOCKS
CHECK OUT: HOST j
TIMESTAMP : t2
FREE BLOCKS
CHECK OUT: NONE
TIMESTAMP :
FREE BLOCKS
CHECK OUT: NONE
TIMESTAMP :
628
624
629
625
630
626
631
627

FIG.11

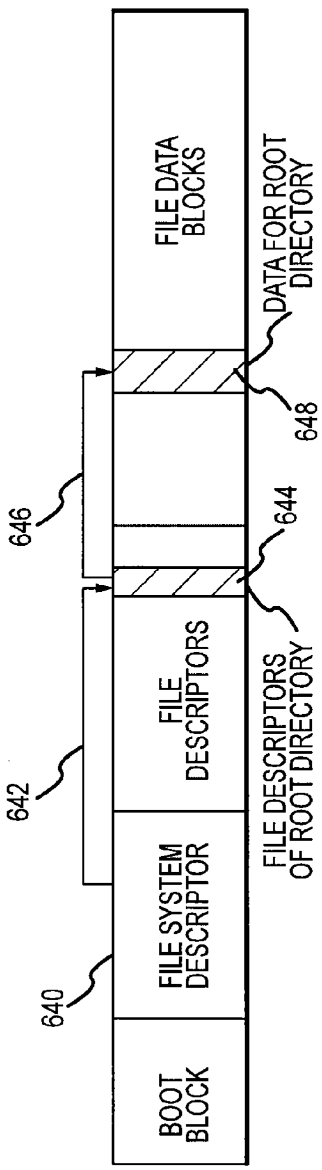


FIG. 12
(PRIOR ART)

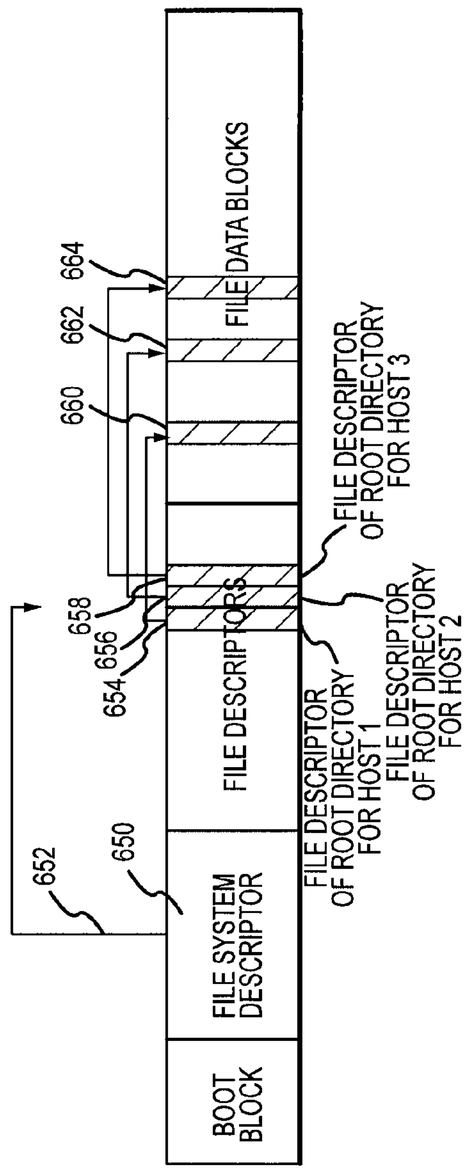


FIG. 13

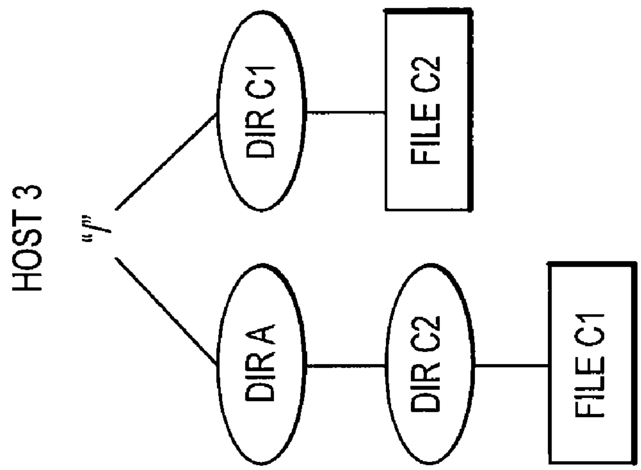


FIG.14C

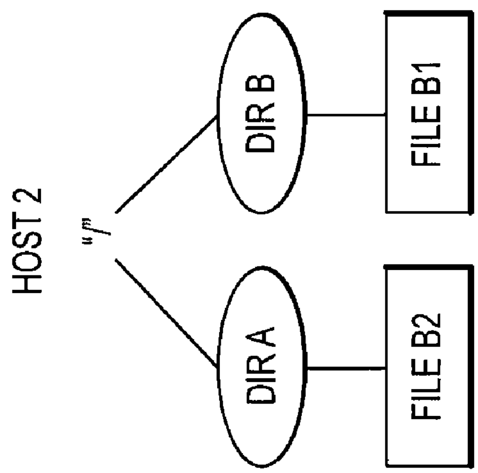


FIG.14B

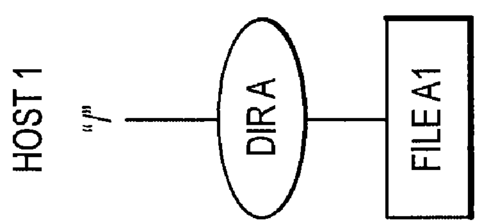


FIG.14A

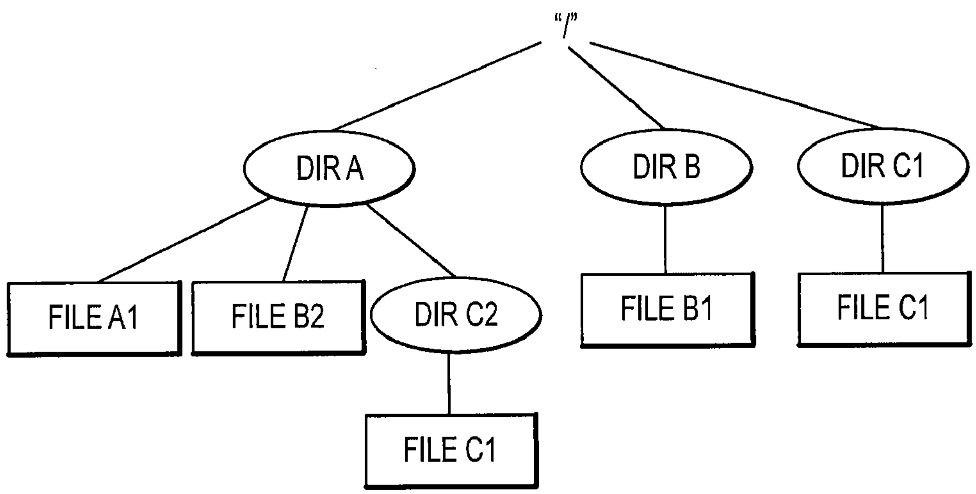

"/"
DIR A
DIR B
DIR C1
FILE A1
FILE B2
DIR C2
FILE B1
FILE C1
FILE C1

FIG.15

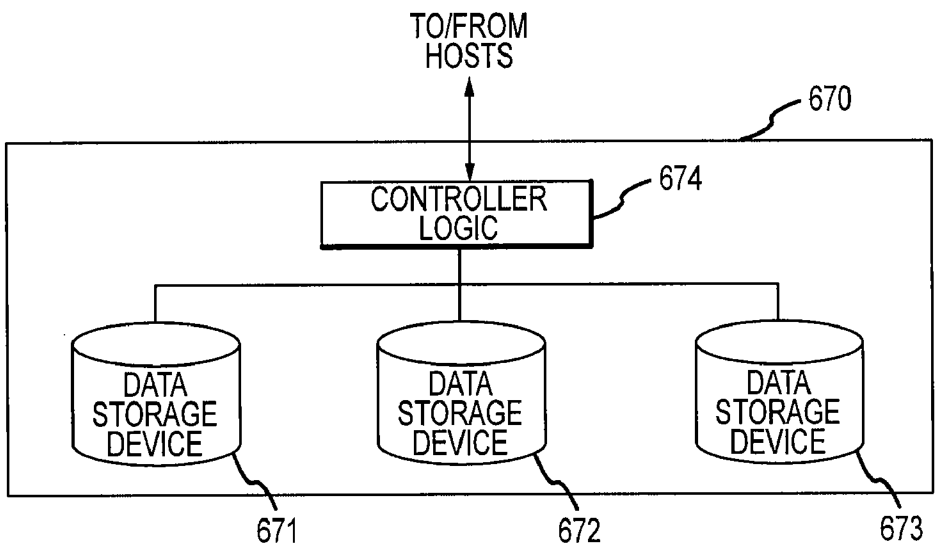
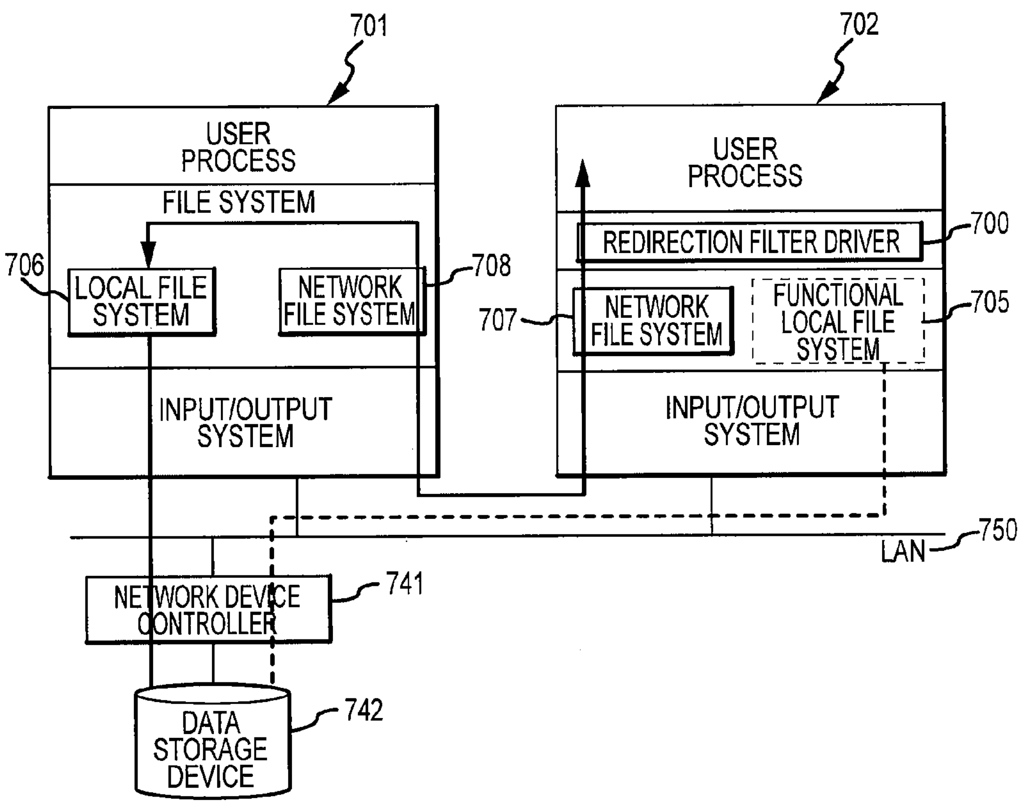

TO/FROM HOSTS
670
674
CONTROLLER LOGIC
DATA STORAGE DEVICE
DATA STORAGE DEVICE
DATA STORAGE DEVICE
671
672
673

FIG.16



701
702
USER PROCESS
FILE SYSTEM
706
LOCAL FILE SYSTEM
NETWORK FILE SYSTEM
708
INPUT/OUTPUT SYSTEM
USER PROCESS
REDIRECTION FILTER DRIVER
700
707
NETWORK FILE SYSTEM
FUNCTIONAL LOCAL FILE SYSTEM
705
INPUT/OUTPUT SYSTEM
LAN
750
NETWORK DEVICE CONTROLLER
741
DATA STORAGE DEVICE
742

FIG.17

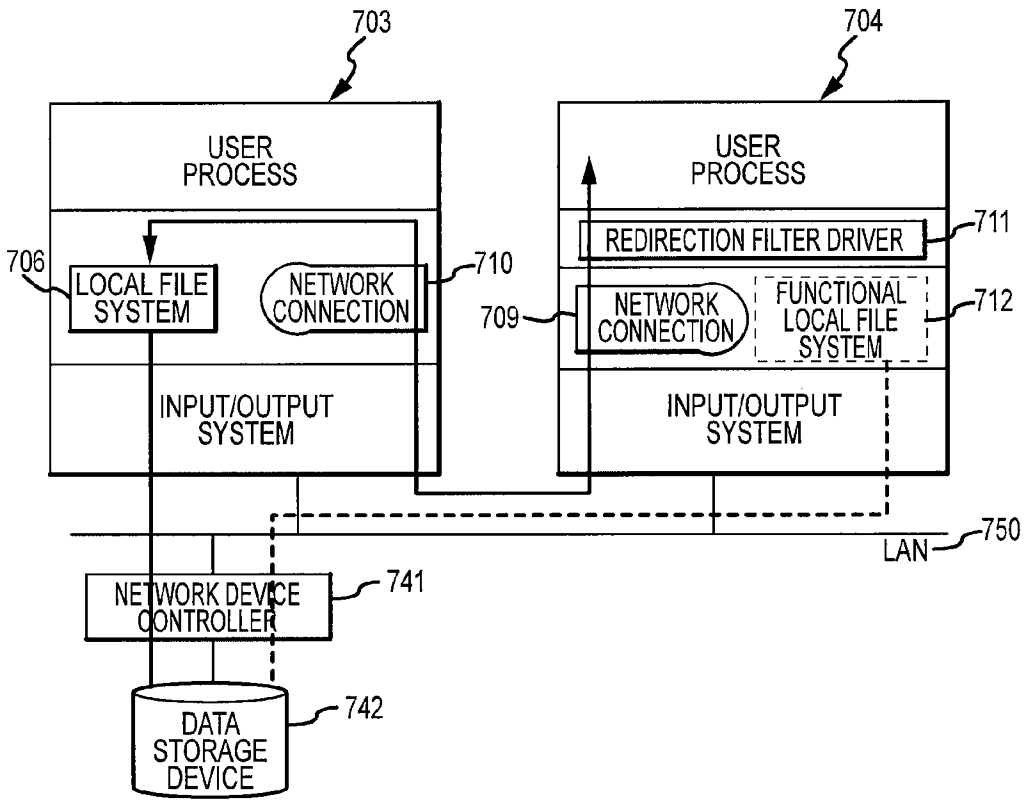

703
704
USER PROCESS
USER PROCESS
REDIRECTION FILTER DRIVER
711
706
LOCAL FILE SYSTEM
NETWORK CONNECTION
710
709
NETWORK CONNECTION
FUNCTIONAL LOCAL FILE SYSTEM
712
INPUT/OUTPUT SYSTEM
INPUT/OUTPUT SYSTEM
LAN
750
NETWORK DEVICE CONTROLLER
741
DATA STORAGE DEVICE
742

FIG.18

SYSTEM USING A SINGLE HOST TO RECEIVE AND REDIRECT ALL FILE ACCESS COMMANDS FOR SHARED DATA STORAGE DEVICE FROM OTHER HOSTS ON A NETWORK

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims the benefit of the following: U.S. Provisional Application Ser. No. 60/506,829, entitled "Device-Level Data Integrity Scheme for Data Devices Shared by Multiple Hosts through LAN," filed Sep. 26, 2003; U.S. Provisional Application Ser. No. 60/590,722, entitled "Low-Level Communication Layers and Device Employing Same," filed Jul. 22, 2004; and U.S. Provisional Application Ser. No. 60/581,691, entitled "File System Features That Enable a Storage Device to Be Shared Directly by Multiple Hosts at Device Level," filed Jun. 21, 2004. These applications are hereby incorporated by reference in their entirety.

BACKGROUND OF THE INVENTION

a. Field of the Invention

The invention relates generally to a data storage device shared by multiple hosts by way of a network. More specifically, the invention relates to efficient access by multiple hosts of a data storage device over a network while maintaining the data integrity of the storage device.

b. Background of the Invention

Generally, hosts, as referred to herein, are electronic devices that employ data storage devices to store digital data for later retrieval and processing by the host. Hosts include, but are not limited to, computers (including desktop personal computers, laptop personal computers and workstations), personal digital assistants (PDAs), digital audio systems, digital television sets, television set-top boxes, digital game devices, smart phones, hand-held computers and other digital data processing devices. Data storage devices include, but are not limited to, hard disk drives, tape drives, flash memory units, and compact disc (CD) and digital versatile disc (DVD) drives. Further, the data written to or read from a storage device may take any of a variety of forms, including, for example, numerical, textual, audio and video data.

Often, data storage devices are not connected directly with a host, but instead communicate with the host via an intermediate electronic device called a device controller, which couples a data storage device with a central processing unit (CPU) and/or logic memory of the host, thus providing a mechanism for transferring data therebetween. Generally, the host also employs its operating system (i.e., the software resident on a host that controls the overall operation of the host) to facilitate communication between the CPU and/or logic memory and the device controller. For example, FIG. **1** depicts a typical hardware configuration for a computer system. A device controller **11** attached to a system bus **9** of a computer system enables data transfers between data devices **12**, **13**, and a CPU **3** and logic memory **4**. As shown in FIG. **1**, the device controller **11** may control one or more data storage devices.

Generally, in situations in which a data storage device is to be shared between two or more hosts, the storage device is not shared among the hosts directly at device level. Instead, the storage device often resides within one of the hosts involved, while the remaining hosts communicate with the data storage device by way of the host containing the data device. Typically, communication between the hosts occurs via a network file system. In general, a file system (for example, the NT File System (NTFS) employed by Microsoft Windows®) is the portion of an operating system responsible for the storage and tracking of files, and a file system that cooperates with other file systems over a network is termed a network file system. Typically, maintaining a stable state in such a system requires that all file write operations by one host, including the writing of any file directory information and other file "meta-data," be allowed to complete prior to allowing access by another host.

However, in such a network a complete computer system or host providing a network file system is required for each data storage device added to the network, thus significantly increasing the cost of the system. More importantly, the multiple hosts possess a file system dependency when sharing data at the file system level, as all of the hosts involved must agree on a particular network file system protocol. Further, when a change in the network file system is required, that change must be implemented in all hosts involved. In addition, the usage of files remotely accessible through network file systems typically is limited compared to what is possible by way of direct access of the files from a data storage device, such as a hard disc drive.

Sharing a storage device directly at the device level through a network provides certain advantages over indirect sharing of the storage device via network file systems. Direct sharing tends to be more efficient in terms of latency and access times. Direct sharing is more cost effective because less expensive hardware, in the form of a network device controller may be used instead of an entire computer system, which allows direct connection of each storage device via a network. No additional operating system or file system software is required, which also eliminates the file system dependency problems and limitations identified above.

However, given that such a system provides no centralized control of data transfers between the hosts and the storage device, data integrity is a potential problem. For example, with each host of the system writing and reading various portions of the storage device, one host may easily overwrite portions of files previously written by another host, thus possibly causing partial or total loss of the data contained in the files.

To further explain, sharing a data storage device over a network presents unique challenges compared to, for example, those involved with sharing a network printer. A network printer is often shared by more than two host computers, but the nature of the data being transferred over the network necessitates the two situations be treated differently.

Print commands from computers to network printers apply only to complete files. As a result, all commands issued to a network printer are guaranteed to be serialized at the file level so that no overlapped or interleaved files may be printed. In other words, a file in the shared network printer environment cannot be divided into smaller portions to be interleaved with portions of other files to be printed.

However, files intended for a data storage device, such as a hard disk drive, are ultimately translated into one or more physical sectors of the data device by way of file system software. Further, no guarantee exists that the file will not occupy several discontinuous series of sectors on the data storage device. Therefore, different files from various hosts sharing the storage device may possibly be mapped onto overlapping sectors unless the file systems of the hosts cooperate in some manner.

Given the foregoing, systems and methods that allow multiple hosts to access a shared data storage device in an efficient manner without loss of data integrity would be advantageous.

BRIEF SUMMARY OF THE INVENTION

Generally, embodiments of the present invention allow read and/or write access by multiple hosts, such as computers or other information appliances, to a data storage device by way of a network while maintaining the data integrity of the data storage device. In one particular embodiment, a method for accessing the data storage device provides, in part, acquiring a resource lock, which provides exclusive access to one of the multiple hosts at a time. The host holding the lock may then directly access the storage device without interference from the other hosts. After accessing the storage device, the accessing host releases the lock on that storage device so that other hosts may then be allowed to access the storage device. The lock may be implemented entirely in software, hardware, or a combination thereof. In one embodiment, the lock is implemented within the data storage device, and the data storage device accepts and executes lock access commands issued by the multiple hosts.

In another embodiment, a networked system is provided which includes a data storage device and a plurality of hosts coupled to the storage device by way of a digital network. In addition, a resource lock is included which provides exclusive access to the data storage device to one of the plurality of hosts at a time. Digital networks employable for coupling the data storage device with the multiple hosts include, but are not restricted to, a local area network such as Ethernet (LAN), a wide area network (WAN), the Internet, a virtual private network (VPN), and any other digital network.

In another embodiment of the invention, a networked system with a data storage device shared by a plurality of hosts over a network utilizes a file system providing a "check out" attribute for each cluster of free blocks available for file storage. A host attempting to claim a cluster of free blocks analyzes the associated check out attribute to determine if another host has already claimed the cluster. If not, the host sets the check out attribute with a value indicating that it has claimed the cluster, thereby providing the host exclusive access to the cluster. In a further embodiment, a resource lock may be employed to protect the access to the check out attribute. Mutually exclusive access to other file system data structures may be provided in a similar fashion.

In a further embodiment, a first host has exclusive direct access to a data storage device, which is accessed by way of the host's local file system over a digital network. A second host requiring access to the data storage device communicates with the first host by way of the digital network. File access requests generated by the second host are redirected away from its own local file system to the first host by a redirection filter driver. The first host maintains direct access to the storage device while the second host communicates with the device through the first host's file system. In one embodiment, the first and second hosts each include a file network system for transferring file access requests from the second to the first host. In another embodiment, each of the first and second hosts employ a network connection (such as a socket connection program) to allow the second host to issue file access requests through the first host.

Additional embodiments and advantages of the invention will be realized by those skilled in the art upon reading the detailed description of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** depicts a block diagram of a typical data storage device connection within a computer system.

FIG. **2** depicts a block diagram of a computer network connecting multiple host computers with a data storage device at the device level.

FIG. **3** depicts a block diagram of a typical data abstraction hierarchy of a computer system wherein each layer of the hierarchy provides a different view of a file.

FIG. **4** depicts a block diagram describing a file access process of a typical computer system employing the data abstraction hierarchy of FIG. **3**.

FIG. **5** depicts a flow diagram describing a typical file read or write operation.

FIG. **6** depicts a flow diagram of a file write operation according to an embodiment of the invention employing a network of multiple hosts sharing a data storage device and a resource lock.

FIG. **7** depicts a flow diagram of a file read operation according to the embodiment of the invention associated with FIG. **6**.

FIG. **8** depicts a typical logical view of a conventional file system on a data storage device.

FIG. **9** depicts a logical view of a Microsoft Windows® NTFS partition.

FIG. **10** depicts a typical network of hosts that share a data storage device directly at the device level.

FIG. **11** depicts a free block list of a file system according to an embodiment of the invention.

FIG. **12** depicts a process of accessing a root directory in a conventional file system.

FIG. **13** depicts a process of accessing multiple root directories of a file system according to an embodiment of the invention.

FIG. **14**A depicts an example of a first partial directory structure of a file system according to an embodiment of the invention.

FIG. **14**B depicts an example of a second partial directory structure of a file system according to an embodiment of the invention.

FIG. **14**C depicts an example of a third partial directory structure of a file system according to an embodiment of the invention.

FIG. **15** depicts the entire directory structure of the examples in FIGS. **14**A, **14**B and **14**C.

FIG. **16** depicts a system employing multiple data storage devices using a file system according to an embodiment of the invention.

FIG. **17** depicts a block diagram of a networked system according to an embodiment of the invention maintaining the data integrity of a data storage device shared by multiple hosts by way of network file systems.

FIG. **18** depicts a block diagram of a networked system according to an embodiment of the invention maintaining the data integrity of a data storage device shared by multiple hosts by way of network connection programs.

DETAILED DESCRIPTION OF THE INVENTION

One embodiment of the invention allows direct connection of one or more hosts to one or more data storage devices, as illustrated in FIG. **2**. A direct connection between a first host **10** and a data storage device (or subsystem) **20** over a network, such as a LAN **50**, (and also between a second host **30** and a third host **40**, and the data storage device **20**) may permit the hosts **10**, **30**, **40** to access the storage device without requiring a server to manage such access. Also, such direct connection allows the hosts to circumvent the use of a network file system, as described above, and access the data storage device at a lower, more efficient level of abstraction.

Typically, the connection between the hosts **10**, **30**, **40** and the data storage device **20** is facilitated by way of a network device controller **31**, **32**, **33** and **34** identified with each host **10**, **30**, **40** and data storage device **20**, respectively. In addition, each host **10**, **30**, **40** normally includes a system bus **21**, **22**, **23**, respectively, with a central processing unit (CPU) **51**, **52**, **53** and logical memory **54**, **55**, **56**, coupled with the network device controller **31**, **32**, **33**, to communicate with the storage device **20**.

FIG. **3** provides a graphical representation of the various levels of abstraction by which data stored on a data storage device may be viewed. At the top of the abstraction hierarchy is a user process **100** (e.g., a user application, an assembly language program, an operating system daemon, or the like) executed on a host computer accessing the storage device **103**, which refers to a file **201** by a file name (such as "MyFile" in FIG. **3**), the file **201** being viewed as a sequence of bytes of arbitrary length. At a lower level of abstraction, a file system **101** of the computer system views that same data as a collection of data sectors **204** within a linear array of "logical sectors" or "blocks" **202**, the blocks **202** of the array typically being numbered from zero up to some maximum sector number. Below that, a software device driver (usually including a computer program having instructions to operate or control the storage device **103**), in conjunction with an input/output system **102** may view the data of the file in a fashion closer to its actual physical configuration **203**, or layout, within the data storage device **103**. Generally, an input/output system, such as the Basic Input/Output System, or "BIOS," of a personal computer, translates operating system calls for access to a data storage device into a form understandable by that device. For example, assuming the use of a hard disk drive as a data storage device, the input/output system **102** may recognize the file as a set of data sectors arranged across one or more disk surfaces, or "platters." Further, each platter is then normally divided into several tracks, or "cylinders," which in turn are typically divided into multiple physical sectors. In many hard disk drives, each logical block **202** corresponds to a physical sector of the drive. Other physical media, such as tape drives, CDs or DVDs, exhibit other physical data sector layouts, all of which are compatible with and embraced by the present invention. Data abstraction hierarchies other than that shown in FIG. **3** are also possible. In fact, any data abstraction hierarchy and/or any data layout currently known or otherwise compatible with digital data storage is encompassed by the present invention.

Most host computer systems also utilize a data buffer, or "cache," normally implemented inside the main logical memory of the computer system and logically residing within the data hierarchy between the file system **101** and the input/output system **102**. The buffer is usually employed to increase system performance by saving a copy of a portion of often-used (or recently used) data stored on the data storage device **103** for future computer CPU or main memory accesses to shorten the time required to access that data. Due to the limited size of the buffer compared to the data storage device **103**, the buffer typically is able to hold only a small percentage of the total amount of data residing in the data storage device **103** at any given time.

FIG. **4** depicts a buffer system **105** placed within the data access hierarchy of FIG. **3**. Shown within the buffer **105** is a copy **106** of a portion of user data **108** residing in the data storage device **103**. However, when the copy **106** is valid (e.g., the copy **106** exactly matches the contents of the corresponding data **108** in the data storage device **103**), all requests to access the corresponding data **108** of the data storage device **103** will instead access the copy **106** in the buffer **105** without directly accessing the data **108** resident on the device **103**.

In the example of FIG. **4**, user data **106** is read from the buffer **105** instead of the data storage device **103** if the user data **106** in the buffer **105** is valid. During write operations, all user data to be written to the data storage device **103** will be copied into either a free space of the buffer **105**, or into an area of the buffer **105** holding a copy of that data. The physical sectors holding the user data **108** of the data storage device **103** are written from the copy **106** in the buffer **105** at a later time, depending on the particular buffer flushing strategy employed. The details of various caching and flushing strategies are well-known in the art, and are not critical to the various embodiments of the invention described herein.

As illustrated in FIG. **4**, the buffer **105** also normally maintains a cache of meta-data **107** corresponding to and describing the copy **106** of user data. Meta-data, such as file descriptors, are data necessary for mapping portions of files **201** to blocks **202** for proper storage and retrieval of file data. Meta-data may include, for example, a file's length and physical location on a data storage device **103**. This information is stored on physical sectors of the data storage device **103** as meta-data **109** associated with the data storage device **103**. Depending on the implementation of the particular file system **101** employed, additional mapping information **110**, such as certain file directories found in the data storage device **103**, may also be cached as meta-data in various data structures of the file system **101** itself.

FIG. **5** shows a generalized method of reading or writing data files (or portions thereof) normally employed by a single host directly connected to a single data storage device, using the data hierarchy of FIG. **4**. In operation **301**, the file system **101** receives a file read or write request from a user process **100**. Following receipt, in operation **302** the file system **101** attempts to access a copy **107** of the meta-data in the buffer **105** that contains the mapping information describing the translation from the file **201** to the corresponding blocks **202**. In operation **303**, the system determines whether the copy **107** of the meta-data stored in the buffer **105** is valid. If the meta-data **107** is valid, operation **305** is executed. If, however, the meta-data **107** is invalid or not present in the buffer **105**, the input/output system **102** executes operation **304** and converts the location of the blocks **202** of the required meta-data into the corresponding location of physical sectors **203** of the data storage device **103**, reads the corresponding meta-data **109** from the physical sectors **203** of the data storage device **103**, and copies the meta-data **109** into the buffer **105**, resulting in a valid copy **107** of the meta-data in the buffer **105**. Optionally, the file system **101** may perform operation **303** once again to ensure the meta-data **107** in buffer **105** is valid.

In operation **305**, with a valid copy **107** of the meta-data now available in the buffer **105**, the file system **101** reads the copy **107**. Continuing with operation **306**, the file system **101** determines whether the requested data access requires a read or write of file data. In the case of a read operation, operation **307** is executed, in which the file system **101** determines the proper blocks **202** of the actual user data desired and attempts to access a valid copy **106** of the user data in the buffer **105**. In operation **308**, the file system **101** determines if the copy **106** of user data is not valid or is nonexistent in the buffer **105**. If the copy **106** of user data is invalid or not present, operation **309** is performed, in which the input/output system **102** converts the location of the blocks **202** holding that data into the corresponding location of physical sectors **203** of the data storage device **103**, reads the user data **108** from the physical sectors **203** of the data storage device **103**, and copies the user data **108** into the buffer **105**, resulting in a valid copy **106** of the user data in the buffer **105**. Operation **308** then may be executed once again to ensure the copy **106** of the user data in the buffer **105** is valid. In operation **310**, the file system **101** then reads the copy **106** of the user data from the buffer **105**, thus completing the read request from the user process **100**.

If, instead, the file system **101** determined in operation **306** that a write operation is involved, operation **311** is executed, in which the file system **101** uses the copy **107** of meta-data previously read from the buffer **105** in operation **305** and transfers the user data **106** and associated meta-data **107** to be written to an appropriate location in the buffer **105**, thus making those portions of the buffer **105** valid. At some later time, when the buffer **105** is to be "flushed" (i.e., data in the buffer **105** is to be written to the data storage device **103**), operation **312** is performed, in which the user data **106** and associated meta-data **107** in the buffer **105** are written to the data storage device **103** as user data **108** and meta-data **109** by way of the input/output system **102**, thereby completing the write operation.

As mentioned earlier, allowing multiple hosts concurrent direct access to the data storage device **103** may cause data integrity problems in both the meta-data **109** and the file data **108** located on the data storage device **103**, as well as any copies **106**, **107** of that data in a buffer **105** of each host. For example, referring to FIG. **2**, one host **10** might be in the process of updating a preexisting file resident on a storage device **20** by way of multiple write operations. Before completion of the update, a second host **30** may read the same file from the data storage device **103**, thus receiving an intermediate and incorrect copy of the file.

In addition, the use of a buffer **105** within each of the hosts exacerbates any potential data integrity problems. For example, if each host is accessing copies **106**, **107** of meta-data and file data from its own buffer **105**, updates to those copies **106**, **107** will not be seen by other hosts until that information is flushed from the buffer **105** and written to the data storage device **103**. Accordingly, each host may be attempting to update the same data file in different ways, completely unaware that multiple, dissimilar copies of the same file exist, thus destroying the data integrity of that file.

To address this problem, one embodiment of the invention involves the use of a resource "lock" to prevent access to the data storage device **103** at the device level by more than one host at any particular time. Generally speaking, the lock is acquired by a host attempting access the storage device, including any reading or writing of a data file to the device **103**, and is released after the access operation has been completed. In most embodiments, completion of a write command would include the host in possession of the lock flushing the contents of its buffer **105**, thus ensuring the meta-data and file data of the data storage device **103** has been updated. Only one host may possess the lock at any one time, thereby prohibiting access to the data storage device by any other host. The lock may also be implemented as a "semaphore" or similar construct known in the art. Generally, a semaphore is a flag or similar indicator that is writable and readable by one or more hosts, and is used to relay a simple message between those hosts.

The lock itself may be implemented in several different ways. The lock may be implemented entirely in software (such as device driver or network protocol), although hardware implementations are possible as well, as are hybrid hardware/software implementations. In one embodiment, the data storage device **103** itself may internally store the value of the lock for access by each of the hosts using the device **103**. All access and manipulation of the lock by the host would then be controlled, for example, by a device-level controller within or operably connected to the data storage device **103**. In addition to lock control, the device-level controller may process standard device-level commands normally targeted for a data storage device, such as the commands associated with the Small Computer Systems Interface (SCSI) or Integrated Drive Electronics (IDE) interfaces known in the art.

Generally, a device-level controller is implemented by way of an embedded microcontroller system designed and employed to perform tasks specific to the control and maintenance of the associated data storage device, including the processing of device-level commands, as described above. Typically, such a system employs resources, in terms of algorithmic capability, speed and logical memory, minimally sufficient to perform its assigned duties in terms of data storage device control, but not to implement commands beyond device control. (For example, the microcontroller may respond to device-level command, but would not implement and run an operating system.) Accordingly, the term "computer," as is generally employed in the art, does not embrace a device-level controller of the embodiments of the present invention.

In some embodiments, the data storage device **103** (or associated controller) may provide lock control and access for the hosts by way of lock-specific commands that facilitate the checking and setting of the lock. For example, the data storage device **103** may provide an "acquire lock" command that provides both a checking and setting of the lock in one operation. In other words, if the lock is already held by another host, the data storage device **103** will return a failure indication to the requesting host. Otherwise, a success indication can be returned, indicating to the requesting host that the lock acquisition was successful, in which case the requesting host may proceed with accessing data on the data storage device **103**. In another implementation, the requesting host may be able to specify a timeout period as a parameter of the acquire lock command so that the data storage device **103** attempts to acquire the lock for the requesting host for the timeout period if the lock is held by another host before returning a failure indication. In addition, the data storage device **103** would also support a "release lock" command to allow the host holding the lock to allow access by the other hosts. In another embodiment, a network device controller **34**, as shown in FIG. **2**, may implement the locking mechanism instead of the data storage device **103** of FIGS. **3** and **4**.

In yet another implementation, the lock may be implemented by one of the hosts accessing the data storage device **103**. Other hosts would then access the lock by making requests via the network to the host implementing the lock. In another embodiment, responsibility for maintaining the lock may be distributed among all or some subset of the hosts involved, with each host possessing a local copy of the state of the lock, including the identity of the host currently in possession of the lock. In that particular case, care must be taken to ensure that each of the hosts maintains a current copy of the state of the lock.

FIG. **6** generally shows the steps of a write operation of a file according to an embodiment of the invention employing a resource lock to enhance the data integrity of the file being written. Using the system of FIG. **4** as a template for a host and a data storage device, operation **401** is first executed, in which the user process **100** issues a file write request to the file system **101**. In operation **402**, the file system **101** first acquires the lock before initiating the actual write operation. As stated above, acquisition of the lock may be delayed if another host is already in possession of the lock. In one embodiment, the requesting host may "time out" if the lock has not been acquired within a predetermined period of time, thus allowing the host to complete other tasks before attempting to acquire the lock once again. During operation **402**, the file system **101** may intermittently check for the lock explicitly, or such an intermittent check may be performed automatically by another software routine. Alternately, the file system may place a request for the lock, and then be interrupted by another software routine when the lock becomes available.

Once the lock is acquired, operation **403** is performed, in which the file system **101** checks for a valid copy **107** of the meta-data in the buffer **105**. In some cases, the file system **101** may assume beforehand that the meta-data **107** in the buffer **105** is invalid if the previous lock holder was another host, thereby circumventing an exhaustive check of the buffer **105**. In such a case, an identification of the last host to hold the lock may also be implemented in conjunction with the lock itself. For example, a host may write a specific file system data structure on the data storage device **103** after acquiring a lock indicating it is the most recent holder of the lock.

If the meta-data in the buffer **105** is not present or valid, operation **404** is executed, in which the corresponding meta-data **109** is read from the data storage device **103** (i.e., the data storage device **103**) using the input/output system **102**. Otherwise, the file system **101** may proceed to operation **405**. Once operation **404** is complete, the file system **101** may again check if validity of the meta-data in the buffer **105** is valid in operation **403**. With a valid copy **107** of the meta-data now in the buffer **105**, operation **405** is performed, in which the file system **101** reads that copy **107** of the meta-data, translates the information for the file **201** into a set of blocks **202** containing the desired user data, and then writes the new data into the proper location in the buffer **105**. In addition, the meta-data is updated based on any changes necessary due to the new user data to be written.

The file system **101** then determines in operation **406** if the host will hold the lock after the write operation. If not, in operation **408** the host flushes all file data and meta-data in the buffer **105** to the data storage device **103** using the input/output system **102** to perform the logical-to-physical sector translation before it releases the lock in operation **409**. If the host holds the lock, the host then determines in operation **407** if another write request is pending. If so, the write procedure resumes by returning to operation **403**. If there is no pending write request, the host determines if it will still hold the lock in operation **406** based on other criteria, such as anticipated near-term data storage device access requirements.

As a result, the lock mechanism prevents multiple hosts from simultaneously or concurrently writing to the data storage device **103** by allowing only one host at any time to hold the lock for writing operations. Accordingly, the writing by any host holding a lock is likely valid, because all previous write operations to the shared data storage device **103** by other hosts are fully recorded by flushing all user data **106** and meta-data **107** from the buffer **105** to the data storage device **103** before another host can acquire a lock.

In an alternative embodiment, all hosts connected to the data storage device **103** may cooperate to time-share access to the data storage device **103** by allocating a limited lock hold time to each host, thus preventing monopolization of the lock by any single host. The limited hold time may be invariant (for example, ten seconds per host), prioritized by host (for example, host A has a ten second lock while host B has a five second lock), or varying with some other parameter (for example, larger files may permit longer locks, lock duration may be determined by data throughput speed of the storage device and the host, and so forth).

FIG. **7** illustrates the process flow of a read operation, according to another embodiment of the invention, utilizing a lock to guarantee the file data read is consistent with the data resident on the data storage device **103**. In operation **501**, the file system receives a read request from a user process **100**. Thus, in operation **502**, the file system **101** acquires a lock before initiating a read operation. Once the file system **101** acquires the lock, operation **503** is executed, in which the file system **101** determines if the copy **107** of meta-data and other mapping information is present and valid in the buffer **105**. If not, in operation **504** the file system **101** reads the corresponding meta-data **109** from the data storage device **103** by way of the input/output system **102** into the buffer **105**. The file system **101** may then return to operation **503** to ensure the validity of the meta-data **107** in the buffer **105**. In operation **505**, the file system **101** reads the copy **107** of the valid meta-data from the buffer **105**, performs a translation from the name of the file **201** to a set of blocks **202**, and searches the buffer **105** for a valid copy **106** of the file data corresponding to the blocks **202**. If a copy **106** of the file data is not in the buffer **105**, or such a copy is not valid, the file system **101** performs operation **506**, in which the file data **108** is read from the data storage device **103**. Again, at this point the file system **101** may return to operation **505** to ensure the validity of the file data **106** in the buffer **105**. With a valid copy **106** of the file data in the buffer **105**, the file system **101** executes operation **507** and completes the read request by obtaining the data copy **106** from the buffer **105**, before releasing the lock in operation **508**. Since the file system **101** has not written any meta-data or file data to the buffer **105**, flushing the buffer prior to releasing the lock is not required, as was the case during the write operation described in FIG. **6**. In alternative embodiments, the file system **101** may retain the lock to execute further read or write operations prior to releasing the lock.

In one implementation, the reading procedures of FIG. **7** may be performed while bypassing the procedures of acquiring and releasing the lock (steps **502**, **508**) if the meta-data **109** and file data **108** are read directly from the data storage device **103** instead of the buffer **105**. If so, any problems with the contents of the buffer **105** remaining consistent with the data storage device **103** are eliminated. However, in some cases, the data read by a host that has not acquired a lock may be in a partially-modified state if another host holding the lock is in the process of writing the same file to the data storage device **103**. As a result, the adequacy of reading data without performing the locking and unlocking operations may be determined by whether data that has been partially-modified is acceptable for host use. In addition, a time-share lock scheme may be implement for the read operation of FIG. **7**, as presented above in the discussion of FIG. **6**.

The write and read operations shown in FIG. **6** and FIG. **7**, respectively, can be implemented by adding filter driver software atop the existing file system **101** without modifying the existing file system or adding an extra file system. Effectively, the filter driver may be layered into or atop the file and/or operating systems. In one embodiment, the filter driver software intercepts the read and write requests to the existing file system software **101** before it acquires the lock. For example, the filter driver software may verify the validity of the meta-data **107** and user data **106** in the buffer system **105** before it notifies the file system **101** about the validity, as well as perform the locking and unlocking procedure.

In another embodiment of the invention, the data integrity of a data storage device shared among multiple hosts over a network at the device level is maintained by a set of features or data structures provided by the file systems of the hosts. These structures provide a mechanism by which free blocks (i.e., blocks not currently written with valid user data or meta-data) of the data storage device are allocated to each host in a mutually exclusive manner. Considering this particular embodiment of the invention involves enhancements to the typical file system currently available, a more detailed discussion of file systems and their operation is provided.

A file system, which typically is the largest unit of data structure identified with a persistent data storage device, normally includes a collection of files, file descriptors, directories and other file-system-related information. The file system is stored on a "logical disk," which can be a physical disk (e.g., a hard disk drive), a disk partition (i.e., some identified portion of a physical disk), several physical disks, or some other physical manifestation of a data storage device. From the file system's standpoint, a logical disk is composed of a large, one-dimensional array of logical blocks, as described above.

FIG. **8** presents an example disk block layout of a file system. Typically, the first block is reserved for a boot block **601**, which is the first disk block read by a computer system to determine where on the disk to retrieve the operating system and other necessary information to initialize the computer after a reset has occurred. A file system descriptor **602** (often called a "super block" in the UNIX operating system) contains information about the file system as a whole, since the file system is viewed as an object or entity in and of itself, and thus requires a data structure to represent it. The file system descriptor **602** defines the total size of the file system in blocks, the size of the file descriptor area, the location of the root directory (i.e., the top directory in a hierarchically-arranged file structure), and other file system meta-data. One other important item of information defined in the file system descriptor **602** is the first block of the free block list.

File descriptors **603** (often called "inodes" in UNIX) contain all meta-data associated with a particular file, including the actual block addresses on the data storage device where the file data is located. Other meta-data in the file descriptors **603** normally include the owner of the file, file protection information, time of creation, time of last modification, time of last use, and other information related to the specific file.

In most file system implementations, directories, which essentially are logical collections of files and other directories, are actually implemented as files, so each directory will have a file descriptor in the file descriptor area and occupy some data blocks in the data block area **604**. In other words, no special areas of the storage device are reserved within the file system for directories.

Generally, at some point in time, some plurality of the data blocks on a data storage device are allocated or occupied by file data and file descriptors, while some blocks remain unallocated, or "free." All allocated blocks are linked to a file descriptor so that file data can be traced by starting with the corresponding file descriptor. As files grow, more free blocks are allocated to the files as necessary. Accordingly, file systems typically maintain a list of free blocks for quick allocation of those blocks to files.

FIG. **8** shows a logical view of a typical file system. The actual locations of each entity depicted may be allocated across the entire allotment of disk blocks. For example, the file descriptors may be located in any of a number of areas on the disk, as they may be positioned with the file names in the directories, in a special area of the disk reserved for file descriptors, or among the disk blocks containing the actual file data.

Different file systems may also define varying structures containing the information required to implement the file system. For example, FIG. **9** depicts a logical view of a partition of the Microsoft Windows® NT File System (NTFS), in which information on the file system itself and the starting block number of the Master File Table (MFT) **606** is stored in a boot sector **605**. The Master File Table **606** holds meta-data concerning every other file and directory contained in the file data blocks **607** of the NTFS partition. The Master File Table **606** also contains sixteen entries, or records, reserved for various special meta-data files, including a free block list.

Traditional file systems are designed such that all free blocks are managed by a single host because the data storage device is normally attached to the inner system bus of the host. As a result, only the single host may directly access the device. However, if two different hosts are able to access the device directly at the device level, as would be the case when a storage device is coupled with the hosts directly via a network, each host may allocate logical blocks from the same free block list independently, thus potentially allocating identical blocks to different files. This, in turn, would corrupt the consistency and integrity of the entire file system.

FIG. **10** shows an exemplary system of multiple host computers **610**, **611**, **612** sharing a single shared data storage device **630** over a network **620** directly at the device level, while relying solely on the local file systems of the hosts **610**, **611**, **612** to manage the storage device **630**. Sharing a hard disk drive or other data storage device directly at the device level is distinguished from sharing a storage device by multiple hosts indirectly via a separate server through a distributed file system by way of the data integrity support provided by a distributed file system. Direct accessing of a remote hard disk drive by one or more computing devices is discussed in U.S. Provisional Application Ser. No. 60/592,722, entitled "Low-Level Communication Layers and Device Employing Same," incorporated by reference herein in its entirety.

For example, some distributed file systems, such as xFS ("x" File System) and GFS (Global File System) utilize a server for managing the meta-data of the file system, including the free block list. Each client host consults the server before the client acquires free blocks for writing a file. The server allocates free blocks, but does not control actual data written to the storage device. This mechanism allows the clients to write and read data blocks onto and from the data storage device directly without relaying the user data to and from the server. However, this mechanism does not allow the hosts to share the storage device directly at the device level without server intervention. Thus, since only the server controls the allocation of free blocks, and each client host must consult the server to obtain free blocks, the file system cannot be corrupted on the basis of allocating free blocks to multiple clients. However, such a mechanism suffers from scalability and performance overhead limitations since a single server intervenes in all free block allocation to provide proper meta-data management and cache coherency. Thus, computer networks employing a distributed file system generally lack the performance associated with a network in which multiple computers access a shared data storage device directly at the device level without the assistance of a file server.

In order for multiple hosts sharing the same storage device to maintain free blocks of the storage device in a manner such that no block is allocated to more than one particular file at a time, the total number of free blocks may be divided into multiple, mutually exclusive sets of free blocks, with each set being accessible to only one particular host at a time. For example, FIG. **11** shows a logical view of a list of free blocks according to various embodiments of the invention. Each of a first, second, third and fourth entry **620**, **621**, **622**, **623** of the free block list, with each entry denoting a "cluster," or mutually exclusive set, of free blocks. Therefore, each free block is included in only one specific free block cluster. Although FIG. **11** depicts the logical view of the entries denoting the free blocks in the form of a list, alternative embodiments of the invention are not limited to a specific data structure, such as a list structure. For example, in many file system implementations entries of free blocks are maintained as a form of "bitmap," wherein each bit represents a cluster on the physical disk, thus identifying whether the cluster is free or has been allocated to a file.

In further reference to FIG. **11**, each free block cluster is associated with a "check out" attribute **624**, **625**, **626**, **627**. Each host computer can check out the mutually exclusive free block clusters only when the clusters are not currently checked out by another host. When a host checks out or claims one or more free block clusters, the host is responsible for setting the check out attribute **624**, **625**, **626**, **627** for each claimed cluster using its host ID. To provide additional security, some embodiments may employ a resource lock (as described above) in the course of setting the check out attribute **624**, **625**, **626**, **627** to prevent a race condition among hosts contending for the same free blocks. Once a host checks out a set of free block clusters, the host may allocate to a file any of the free blocks from the free block clusters that have been checked out since those blocks are reserved exclusively for the use of the host by way of the check out process, thus preventing any other host from checking out those same blocks.

Any checked out blocks not ultimately allocated to a file are subsequently returned by the accessing host to the free block list. To return the unallocated free blocks, the host inserts new nodes of free block clusters into the free block list and leaves the check out attribute **624**, **625**, **626**, **627** of the newly inserted cluster nodes blank.

In addition to setting the check out attribute, the accessing host also sets a timestamp attribute **628**, **629**, **630**, **631** with a value indicating when the free blocks were checked out. The timestamp **628**, **629**, **630**, **631** is employed to prevent a host from holding unallocated free blocks indefinitely. This situation can occur, for example, when a host has checked out one or more clusters of free blocks and then becomes inoperative, or "crashes," before the host has the opportunity to return the unallocated blocks to the free block list. If the timestamp **628**, **629**, **630**, **631** is older than some predetermined value, other hosts may then claim the unallocated blocks from the outdated cluster of free blocks using the standard check out procedure.

In the specific example of FIG. **11**, the first cluster of free blocks denoted by entry **620** is checked out to a host i with a timestamp value **628** of t_1, and the second cluster indicated by entry **621** is checked out to a host j with a timestamp value **629** of t_2. The other clusters denoted by the entries **622**, **623** shown are not checked out, and thus remain available to any host.

The file system stores within its file system descriptor the location of the file descriptor of the root directory. As the root directory and subdirectories accumulate files and other directories, links are provided within each directory pointing to block locations in the file data block area where the files and file descriptors associated with the directory are stored. Therefore, the file system can trace the entirety of the directory structure starting from the root directory. FIG. **12** illustrates how a directory structure is implemented in an exemplary conventional local file system. The file system may follow a link **642** from its file system descriptor **640** to the location of the file descriptor **644** of the root directory, which in turn contains another link **646** indicating the location of additional data **648** associated with the root directory.

In one embodiment of the invention, a separate set of file descriptors for the root directory are provided for each host that shares the device, in contrast to the single set of file descriptors normally employed. FIG. **13** illustrates one possible implementation of multiple file descriptors for the root directory, one for each host that shares a data storage device. In this example, the file system descriptor **650** contains a link **652** to a set of root directory file descriptors **654**, **656**, **658**, each of which is the file descriptor of the root directory for each of three hosts sharing the device.

When a host accesses the file system to view the directory structure, the host peruses the entire directory structure by following the links to the corresponding file descriptors and data starting from the complete set of root directory file descriptors **654**, **656**, **658** allocated to the hosts. In the particular example of FIG. **13**, the links **652** in the file system descriptor **650** point to three root directory file descriptors **654**, **656**, **658**, each of which corresponds to one of the three hosts. Also, each of the root directory file descriptors **654**, **656**, **658**, in turn, points to the corresponding data blocks **660**, **662**, **664** for storing directory information for each of the three hosts, respectively.

In further reference to the exemplary file directory structure depicted in FIG. **13**, presume Host **1** creates a directory "dir A" under the root directory, and creates a file "file A**1**" under the "dir A" directory, as depicted in FIG. **14**A. Host **1** stores information necessary for the directory structure it created in the blocks allocated for its root directory file descriptor **654** and associated data blocks **660**, as shown in FIG. **13**. Similarly, as shown in FIG. **14**B, Host **2** creates its own directory "dir B" under the root directory, and creates a file "file B**1**" under "dir B," using its root directory file descriptor **656** and data blocks **662**. Host **2** then creates another file "file B**2**" under "dir A" originally created by Host **1**, employing information in the root directory file descriptor **654** and data blocks **660** created by Host **1**. Further, as shown in FIG. **14**C, Host **3** creates its own directories "dir C**1**" and "dir C**2**" under the root directory and "dir A" by employing its root directory file descriptor **658** and data blocks **664**. Host **3** then creates files, "file C**1**" and "file C**2**" under "dir A" and "dir C**1**", respectively.

As is evident from the foregoing discussion, any host can read directory structures created by other hosts in order to obtain a complete view of the entire directory structure by following the links starting from the root directory file descriptors allocated for the hosts involved. As a result, any host can create its own files under directories created by other hosts.

Further, if the links starting from one of the root directory file descriptors identified with a particular host (for example, the root directory file descriptor for Host **1**) are followed, the directory structure created by that host may be retrieved. In other words, each of the directory structures created by each host can be retrieved by following the links starting from the root directory file descriptor allocated for the corresponding host.

Based on the foregoing, the directory structure created by each host constitutes a portion of the total directory structure of the entire file system. The total directory structure may thus be obtained by superimposing the partial directory structures created by each of the hosts. For example, FIG. **15** depicts the total directory structure of the data storage device by superimposing the three partial directory structures created by the three hosts, as shown in FIGS. **14**A, **14**B and **14**C.

The file system maintains attributes of a file, as well as the file data itself, to represent related information necessary for management of the file. One such attribute in one embodiment indicates ownership of the file. Because embodiments of the invention allow multiple independent hosts to share the same device directly at the device level, the file system may maintain an "ownership" attribute within the meta-data of each file to distinguish which host maintains ownership of the file. This host identification prevents a host from exercising impertinent access rights to files owned by other hosts.

Another file attribute in another embodiment of the invention is a "check out" attribute of a file, which is distinguished from the "check out" attribute associated with each free block cluster, described in detail above. When a host having write permission for a particular file accesses that file, the file system marks the check out attribute of the file with an identification of the accessing host. At that point, other hosts may not check out the file with write permission. This mechanism prevents more than one host from writing the same file at the same time, which would likely corrupt the data in the file. However, in some embodiments hosts may read a file that is currently checked out by another host holding write permission.

In one embodiment, the file system may require a host to acquire a resource lock before it can check out the file in order to prevent race conditions created by multiple hosts vying to check out the same file. Alternatively, if the data integrity level of a file is relaxed, use of a resource lock can be avoided. For example, some video and audio data files may remain viable even if the data integrity of the files has been compromised to a degree.

Conventional file systems typically maintain in-memory data structures, instead of on-disk data structures (such as the file check out attribute described above), for managing the consistency of files opened by system processes of the host. Storing such data structures in volatile memory, like many forms of random access memory (RAM) (or other non-persistent storage devices), may be appropriate in an environment in which a single host possesses exclusive control of the storage device. In the embodiments described herein, however, multiple hosts may share control of the data storage device. Therefore, data structures relevant to file consistency management that are maintained only within the volatile memories of each host may have limited utility where multiple hosts share the same storage device directly at the device level unless the hosts share the in-memory data structures spread over multiple hosts.

A file system generally maintains files containing information for effective handling of the blocks of the storage device on which the file system is implemented. These files are termed "file system files." Typically, two of these files are the free block file and the bad block file. The free block file contains a list of blocks that are free, and possibly a complementary list of blocks that are written. The bad block file contains a list of "bad," or defective, blocks on which no data can be written to or read from correctly, generally due to defective portions of the recordable medium within the data storage device. Conventional file systems have a single host manage those particular file system files.

In various embodiments of the present invention, each host of a multiple-host system accesses the free block file in a mutually exclusive manner by way of the cluster check out attribute, as described in detail earlier. In addition, resource lock acquisition may be required prior to setting the check out attribute in some embodiments, thereby providing additional data integrity for the free block file. Access to the bad block file may be regulated in a similar fashion so that hosts may add bad blocks to the bad block list in a secure manner as they are encountered in the course of disk operation. Further, the same control mechanism may be applied to maintain the data integrity of any file system files.

Ordinarily, the use of multiple hosts sharing a single data storage device would produce the possibility of file name conflicts among files created by the multiple hosts. For example, if the hosts run the same operating system and use default swap files set automatically by the operating system, two or more swap files associated with different hosts could have the same file name under the same directory. However, in embodiments of the invention, the host ownership attribute (described above) may be utilized to distinguish such files.

The file system features presented herein may also be implemented in a system employing multiple shared storage devices. FIG. **16** presents a data storage system **670** containing multiple data storage devices **671**, **672**, **673** (such as hard disk drives) with controller logic **674**. The controller logic **674** oversees all three data storage devices **671**, **672**, **673** and collectively treats the devices **671**, **672**, **673** as an array of disk blocks of a larger data storage device, thus providing a view of a single data storage device to the hosts that share the system **670**. Alternatively, multiple data storage device partitions of one or more physical storage devices may be presented in a similar manner.

The controller logic **674** may be implemented as hardware, software, or some combination thereof. In one embodiment, a combination of hardware and software could be employed to process device-level commands for each of the data storage devices **671**, **672**, **673**, such as the device-level controller discussed earlier. (For example, the aforementioned device-level controller may implement, or serve as an example of, the controller logic **674**.) Alternatively, the controller logic **674** may be a software driver executed by a microcontroller system capable of transforming and/or relaying device-level commands received from a host to one of the data storage devices **671**, **672**, **673**. In that case, the driver may determine which of the three storage devices **671**, **672**, **673** is the target of the command received and performs any block location translation necessary from the addressing scheme utilized by the host to the scheme employed by the particular storage device **671**, **672**, **673**. The driver would then relay the modified command to the appropriate target storage device **671**, **672**, **673**, which is capable of processing the modified command itself.

In alternative embodiments of the invention, data integrity of a system of multiple hosts sharing a data storage device can be maintained by utilizing file systems already existing in current hosts. More specifically, instead of allowing all hosts to share a data storage device directly at the device level over a network, one host may have access to the data storage device at device level over a network, while all other hosts are allowed indirect access to the data storage device via their network file systems.

FIG. **17** shows an alternative embodiment where only a first host **701** is allowed to mount a networked data storage device **742** (such as a hard disk drive) by way of a network device controller **741** with full read/write privileges onto one of its local file systems **706**. All other hosts requiring access to the data storage device **742**, such as a second host **702**, are not permitted to mount the networked data storage device **742** onto their own local file systems.

In one embodiment using the structure described in FIG. **17**, a redirection filter driver **700**, a software component, resides at the interface between the user process and the file system of the second host **702**. Although the second host **702** has no privilege of directly mounting the data storage device **742**, the redirection filter driver **700** redirects all file access requests from a user process intended for the data storage device **742** toward a network file system **707** of the second host **702**. As a result, the redirection filter driver **700** presents the appearance of a local file system (shown as a functional local file system **705** in FIG. **17**) to the user process as if the data storage device **742** were mounted onto the second host **702**. The network file system **707** of the second host **702** is connected by way of the LAN **750** with a network file system **708** of the first host **701** so that the file access requests directed toward the data storage device **742** generated in the second host **702** are routed to a local file system **706** of the first host **701**, onto which the data storage device **742** is actually mounted.

Sharing the data storage device **742** between the first and second hosts **701**, **702** in this manner provides advantages over a system which employs indirect sharing of a data storage device strictly at the network file system level, as described earlier. Although the embodiment of FIG. **17** exploits features of the network file systems **707**, **708** in order to transmit file access requests and replies between the second host **702** and the data storage device **742** while maintaining data integrity, an additional network file system for the data storage device **742** is not required. Instead, the embodiment of FIG. **17** provides hosts a functional view of a local file system onto which the data storage device **742** appears to be directly mounted. As used herein, the term "functional" refers to a duplication of the view that would be shown if the file were accessible on a local storage device, rather than across a network. The data storage device **742** and its files are handled in exactly the same fashion as actual local data storage devices and files. Due to the operation of the redirection filter driver **700**, the hosts exhibiting a functional local file system view cannot distinguish the data storage device **742** shared over the network from the devices actually mounted on their own conventional local file systems. Accordingly, since the files on the data storage device **742** are not viewed as shared through conventional network file systems, but are instead viewed as stored by way of a local file system, file usage limits that would otherwise exist if the files were shared through conventional network file systems are eliminated.

Similarly, when the network file system **708** of the first host **701** later receives a reply from the local file system **706** to be transferred to the network file system **707** of the second host **702**, the network file system **707** of the second host **702** directs the reply to the requesting user process through the redirection filter driver software **700** as though the reply were received from the functional local file system **705**.

If no network file system is available or desirable between a host that physically mounts a data storage device (e.g., the first host **701** of FIG. **17**) and other hosts that do not have direct access to the data storage device (e.g., the second host **702** of FIG. **17**), a network connection program may be utilized instead of a network file system to transfer requests and replies for file accesses involving a data storage device.

FIG. **18** illustrates an alternative embodiment wherein only a first host **703** possesses direct access to a data storage device **742**, such as a hard disk drive, while other hosts, such as a second host **704**, may only access the data storage device **742** via the first host **703**. Each of the first and second hosts **703**, **704** employ a network connection program **710**, **709** for communication between the hosts **703**, **704**, respectively. No network file systems are required. Similar to the system of FIG. **17**, a redirection filter driver **711**, residing between a user process and the file system of the second host **704**, intercepts data storage device access requests from the user process and redirects the requests to the network connection **709** of the second host **704**. This network connection **709** then relays these requests to its counterpart network connection **710** of the first host **703**, which in turn directs the request to a local file system **706** of the first host **703**, onto which the networked data storage device **742** is mounted through a network device controller **741**. As was the case with the embodiment outlined in FIG. **17**, the redirection filter driver **711** operates to present a functional local file system **712** to a user process of the second host **704**, making the fact that the data storage device **742** is not mounted locally to the second host **704** transparent to the user process.

When the network connection **710** of the first host **703** then receives a reply from the local file system **706** to be transferred to its counterpart network connection **709** of the second host **704**, the network connection **709** of the second host **704** directs the reply to the requesting user process through the redirection filter driver software **711** as if the reply were received from the functional local file system **712**. The network connections **709**, **710** can be any program that transfers requests and replies therebetween, such as a conventional socket program.

Certain features of the invention described herein may be implemented as an additional layer in or atop the file and/or operating systems. For example, the aforementioned filter driver software may be added to an existing file system without requiring modification of the file system. Similarly, certain features of the invention may be implemented as an additional attribute of a storage device, storage device controller, or storage device file system/structure. For example, the aforementioned check out attribute may augment a file system or structure to provide added functionality. The basic file system/structure may remain relatively unchanged. In other words, the basic functionality and features of the core file system, structure, operating system, and so forth remain unchanged by the invention, which provides added functionality.

Disclosed herein are several embodiments of systems and methods for ensuring the data integrity of a networked data storage device that is shared among a plurality of hosts. While these embodiments are described in specific terms, other embodiments encompassing principles of the invention are also possible. For example, various features of one embodiment may be combined with features of other embodiments to create a new embodiment not specifically discussed herein. Thus, the scope of the invention is not to be limited to the disclosed embodiments, but is determined by the following claims.

What is claimed is:

1. A networked system for providing access to user data while preserving the data integrity of the user data, comprising:

a data storage device comprising the user data;

a plurality of hosts coupled at a device level to the data storage device by way of a digital network; and

a resource lock providing exclusive access to the data storage device to one of the plurality of hosts at a time;

wherein commands to the data storage device originating from any one of the plurality of hosts are directed to a first host, the first host redirecting all such commands to the data storage device.

2. The networked system of claim **1**, wherein the resource lock is acquired and released by way of software commands issued over the network by the hosts to the data storage device.

3. The networked system of claim **1**, wherein the resource lock is implemented entirely in software.

4. The networked system of claim **1**, wherein the resource lock is released after being held for a predetermined period of time.

5. The networked system of claim **1**, the data storage device comprising a hard disk drive.

6. The networked system of claim **1**, the data storage device comprising a writable compact disc drive.

7. The networked system of claim **1**, the data storage device comprising a writable digital versatile disc drive.

8. The networked system of claim **1**, the data storage device comprising a magnetic tape drive.

9. The networked system of claim **1**, wherein at least one of the plurality of hosts comprises a personal computer.

10. The networked system of claim **1**, wherein at least one of the plurality of hosts comprises a workstation.

11. The networked system of claim **1**, wherein at least one of the plurality of hosts comprises a personal digital assistant.

12. The networked system of claim **1**, wherein the digital network is a local area network.

13. The networked system of claim **1**, wherein the digital network is a wide area network.

14. The networked system of claim **1**, wherein the digital network is the Internet.

15. A networked system for providing access to user data while preserving the data integrity of the user data, comprising:

a data storage device comprising the user data;

a plurality of hosts coupled at a device level to the data storage device by way of a digital network;

a resource lock resident on the storage device and providing exclusive access to the data storage device to one of the plurality of hosts at a time; and

filter driver software in each of the plurality of hosts, the filter driver software configured to intercept read and write requests generated within each of the plurality of hosts intended for the data storage device and forward such requests to a single unique host of the plurality of hosts.

16. A networked system for providing access to user data while preserving the data integrity of the user data, comprising:

a data storage device comprising the user data;

a first host having exclusive direct access to the data storage device by way of a digital network, the first host comprising a local file system coupled with the data storage device; and

a second host coupled to the first host by way of the digital network, the second host comprising a redirection filter driver configured to direct a file access request generated by the second host to the data storage device by way of the local file system of the second host instead of a local file system of the first host.

17. The networked system of claim **16**, wherein:

the first host further comprises a first network file system coupled with the digital network and the local file system of the first host; and

the second host further comprises a second network file system coupled with the digital network and the redirection filter driver of the first host;

the second network file system is configured to receive the file access request from the redirection filter and send the file access request to the first network file system; and

the first network file system is configured to forward the file access request to the data storage device by way of the local file system of the first host.

18. The networked system of claim **16**, wherein:

the first host further comprises a first network connection coupled with the digital network and the local file system of the first host; and

the second host further comprises a second network connection coupled with the digital network and the redirection filter driver of the first host;

the second network connection is configured to receive the file access request from

the redirection filter and send the file access request to the first network connection; and

the first network connection is configured to forward the file access request to the data storage device by way of the local file system of the first host.

19. The networked system of claim **18**, the network connection comprising a socket connection program.

20. The networked system of claim **16**, the data storage device comprising a hard disk drive.

21. The networked system of claim **16**, the data storage device comprising a writable compact disc (CD) drive.

22. The networked system of claim **16**, the data storage device comprising a writable digital versatile disc (DVD) drive.

23. The networked system of claim **16**, the data storage device comprising a magnetic tape drive.

24. The networked system of claim **16**, wherein at least one of the first and second hosts comprises a personal computer.

25. The networked system of claim **16**, wherein at least one of the first and second hosts comprises a workstation.

26. The networked system of claim **16**, wherein at least one of the first and second hosts comprises a personal digital assistant (PDA).

27. The networked system of claim **16**, the digital network comprising a local area network (LAN).

28. The networked system of claim **16**, the digital network comprising a wide area network (WAN).

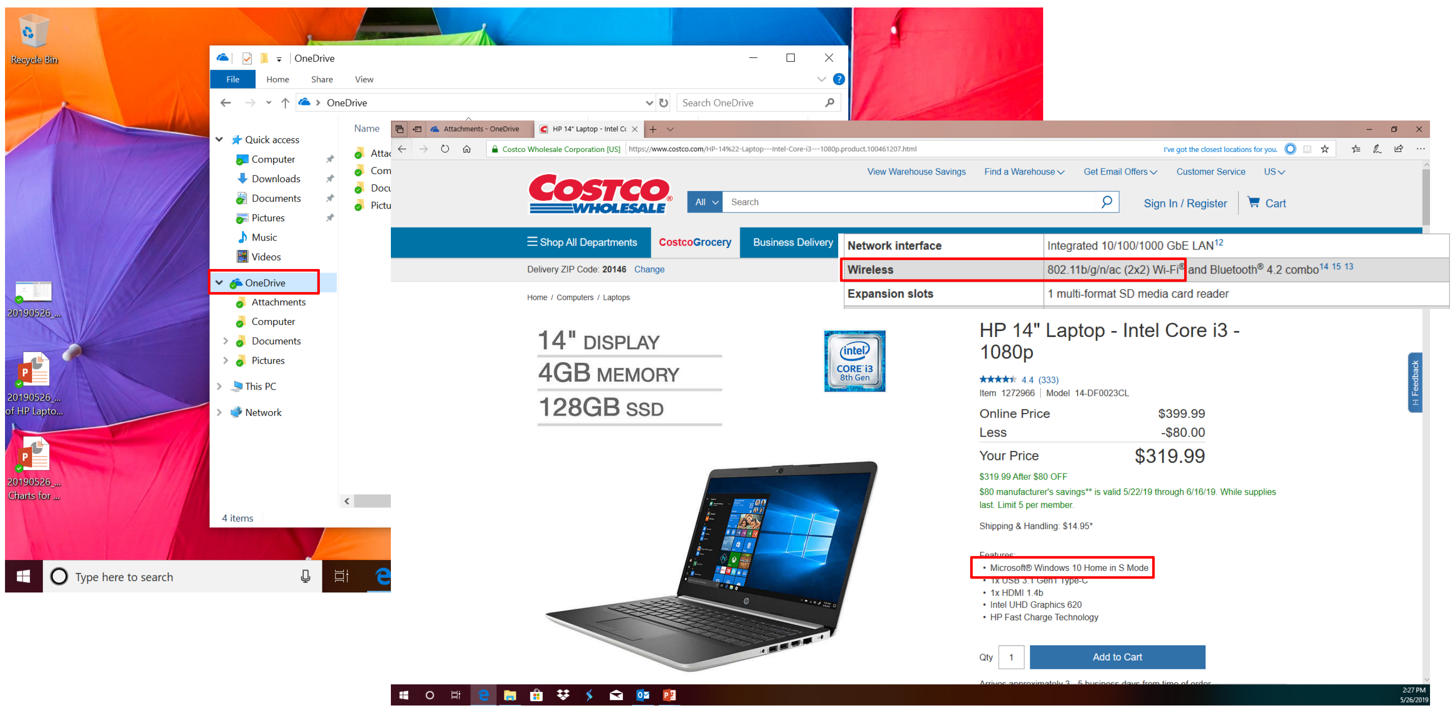
29. The networked system of claim **16**, the digital network comprising the Internet.

* * * * *

EXHIBIT L

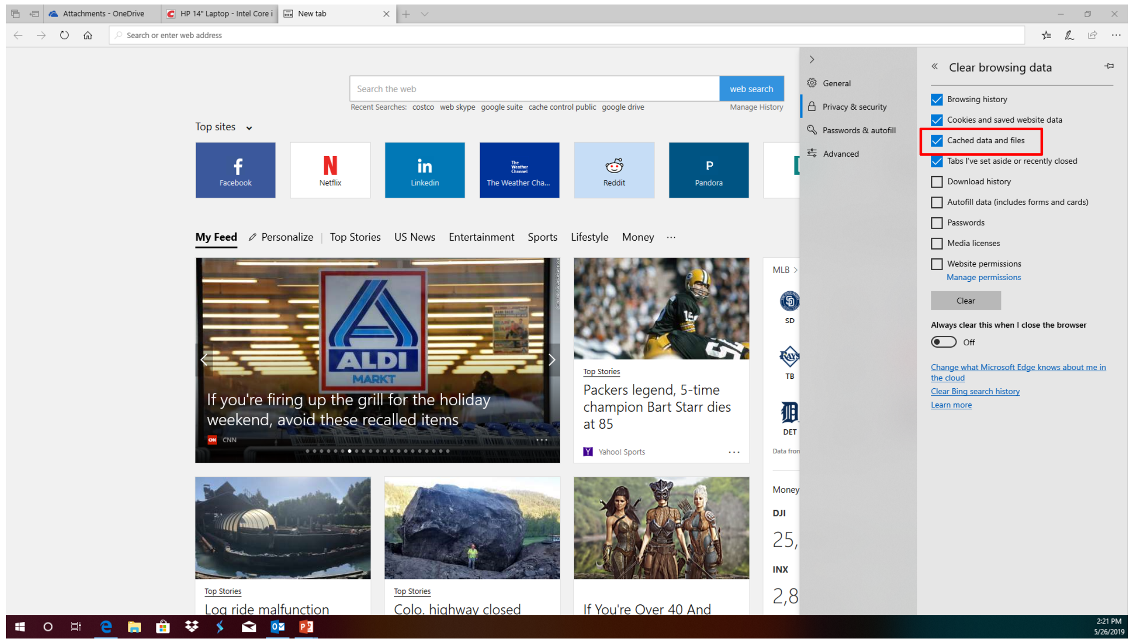
Exhibit 2

Exhibit 2 – Claim Chart for U.S. Patent No. 9,098,526

Claims of the '526 Patent	HP Device Access to Remote Storage Space
1. A wireless device comprising:	A wireless device such as HP Laptop, e.g., HP 14" Laptop – Intel Core i3 with Windows OS and 802.11 b/g/n/ac WiFi, can access a remote storage space provided by storage service providers such as pre-installed Microsoft OneDrive ("OneDrive"), the HP Laptop comprising: 

Case 1:19-cv-01960-RGA Document 15-1 Filed 11/12/19 Page 15 of 120 PageID #: 143

Exhibit 2 – Claim Chart for U.S. Patent No. 9,098,526

Claims of the '526 Patent	HP Device Access to Remote Storage Space
at least one cache storage, one wireless interface, and program code configured to cause the wireless device to:	HP Laptop comprises at least one cache storage, e.g., a cache storage configured for caching data and files such as for the Microsoft Edge web browser; one wireless interface, e.g., 802.11 b/g/n/ac WiFi; and program code, e.g., Windows OS: 

Case 1:19-cv-01360-RGA Document 1-1 Filed 11/12/19 Page 16 of 120 PageID #: 144

Exhibit 2 – Claim Chart for U.S. Patent No. 9,098,526

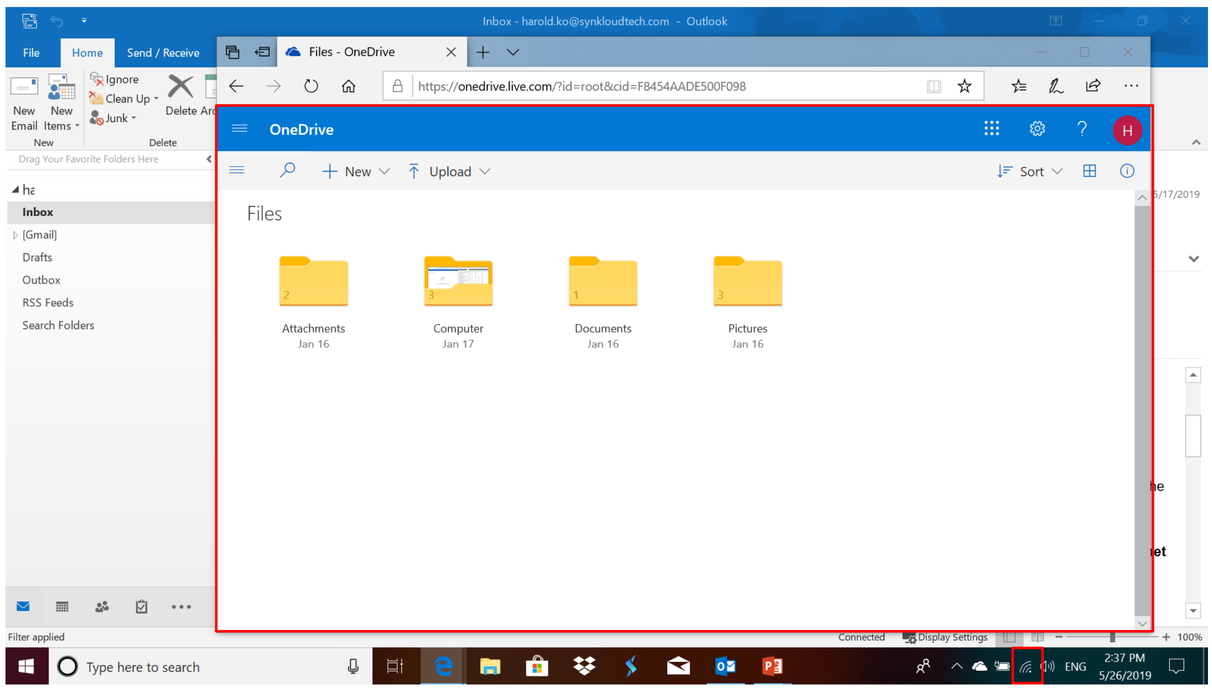
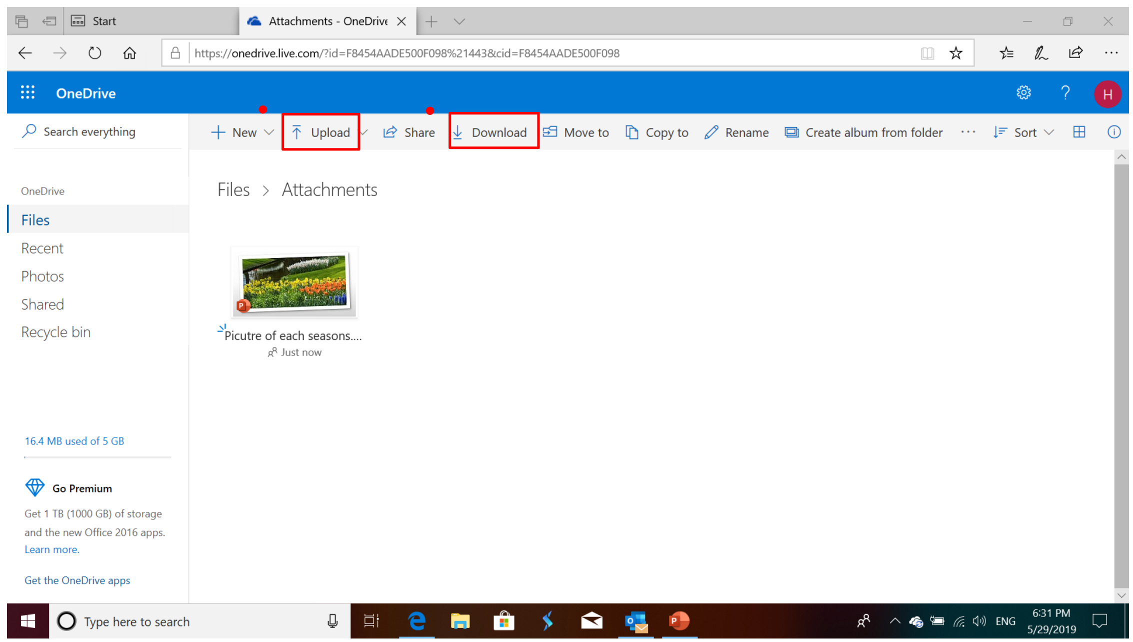
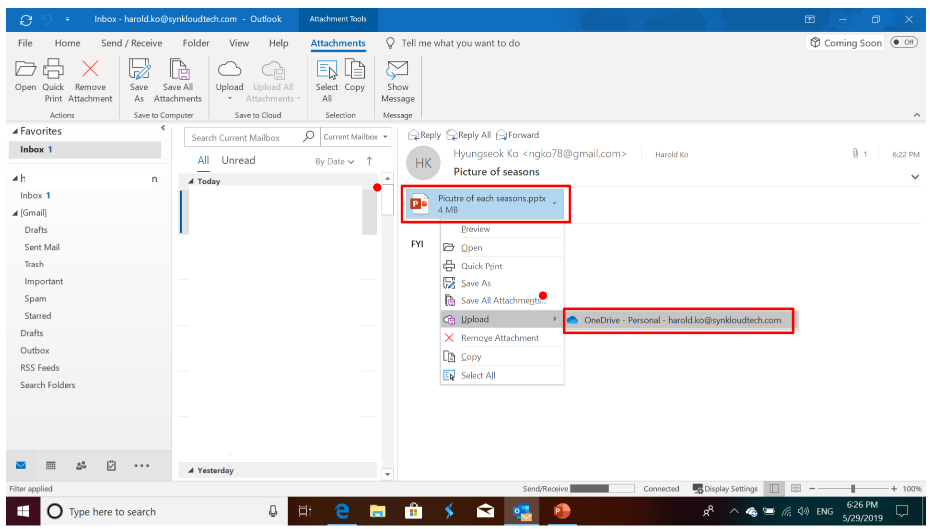
Claims of the '526 Patent	HP Device Access to Remote Storage Space
establish a wireless link for the wireless device accesses to a storage space of a predefined capacity assigned exclusively to a user of the wireless device by a storage server, and	The HP Laptop establishes a wireless link, e.g., WiFi, for the wireless device to access the storage space provided by the OneDrive via Microsoft Edge web browser that is illustrated in a screenshot taken from the HP Laptop. The OneDrive server allocates a 5 GB free storage space exclusively to a user, e.g., only the user can manage or share a data in the storage space. 

Exhibit 2 – Claim Chart for U.S. Patent No. 9,098,526

Claims of the '526 Patent	HP Device Access to Remote Storage Space
couple with the storage server across the wireless link to carry out a requested operation for remote access to this assigned storage space in response to the user from the wireless device performed the operation	The HP Laptop couples with the OneDrive server wirelessly to carry out a requested operation for remotely accessing the assigned storage space. The requested operation for the access to the storage space comprises storing data into the remote storage space or retrieving data from the remote storage space, e.g., uploading a file into the remote storage space (storing), or downloading a file from the remote storage space into the HP Laptop (retrieving):
wherein the operation for the remote access to the assigned storage space comprises storing a data object therein or retrieving a data object therefrom,	

Case 1:19-cv-01360-RGA Document 15-1 Filed 11/12/19 Page 18 of 120 PageID #: 148

Exhibit 2 – Claim Chart for U.S. Patent No. 9,098,526

Claims of the '526 Patent	HP Device Access to Remote Storage Space
the storing of a data object including to download a file from a remote server across a network into the assigned storage space through utilizing download information for the file stored in said cache storage in response to the user from the wireless device performing the operation for downloading the file from the remote server into the assigned storage space.	The storing of a data object includes downloading a file from a remote server, e.g., from a remote Outlook Server, into the user's assigned storage space of the OneDrive by using download information for the file cached in the cache storage of the HP Laptop in response to the user performing the operation of downloading the file. The next screenshot illustrates such downloading a file, e.g., downloading a 2MB PowerPoint file, from an Outlook Server into the user's assigned storage space of OneDrive. • Downloading a PowerPoint file from an Outlook Server to OneDrive [1st step] choose a file to upload and select "Upload to OneDrive" 

Case 1:19-cv-01360-RGA Document 15-1 Filed 11/12/19 Page 19 of 120 PageID #: 147

Exhibit 2 – Claim Chart for U.S. Patent No. 9,098,526

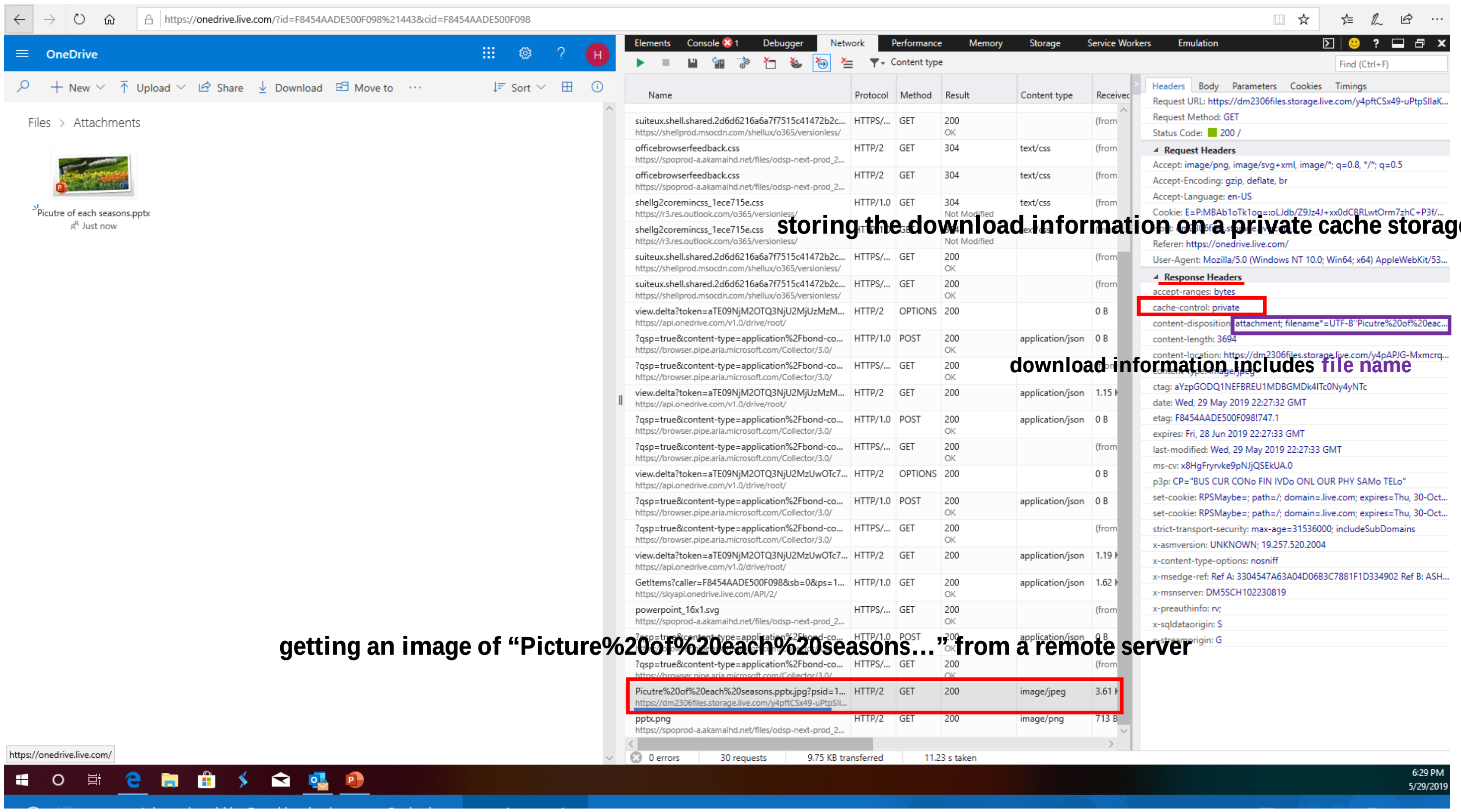
Claims of the '526 Patent	HP Device Access to Remote Storage Space
	[2nd step] utilizing download information for the file stored in the cache storage 

Exhibit 2 – Claim Chart for U.S. Patent No. 9,098,526

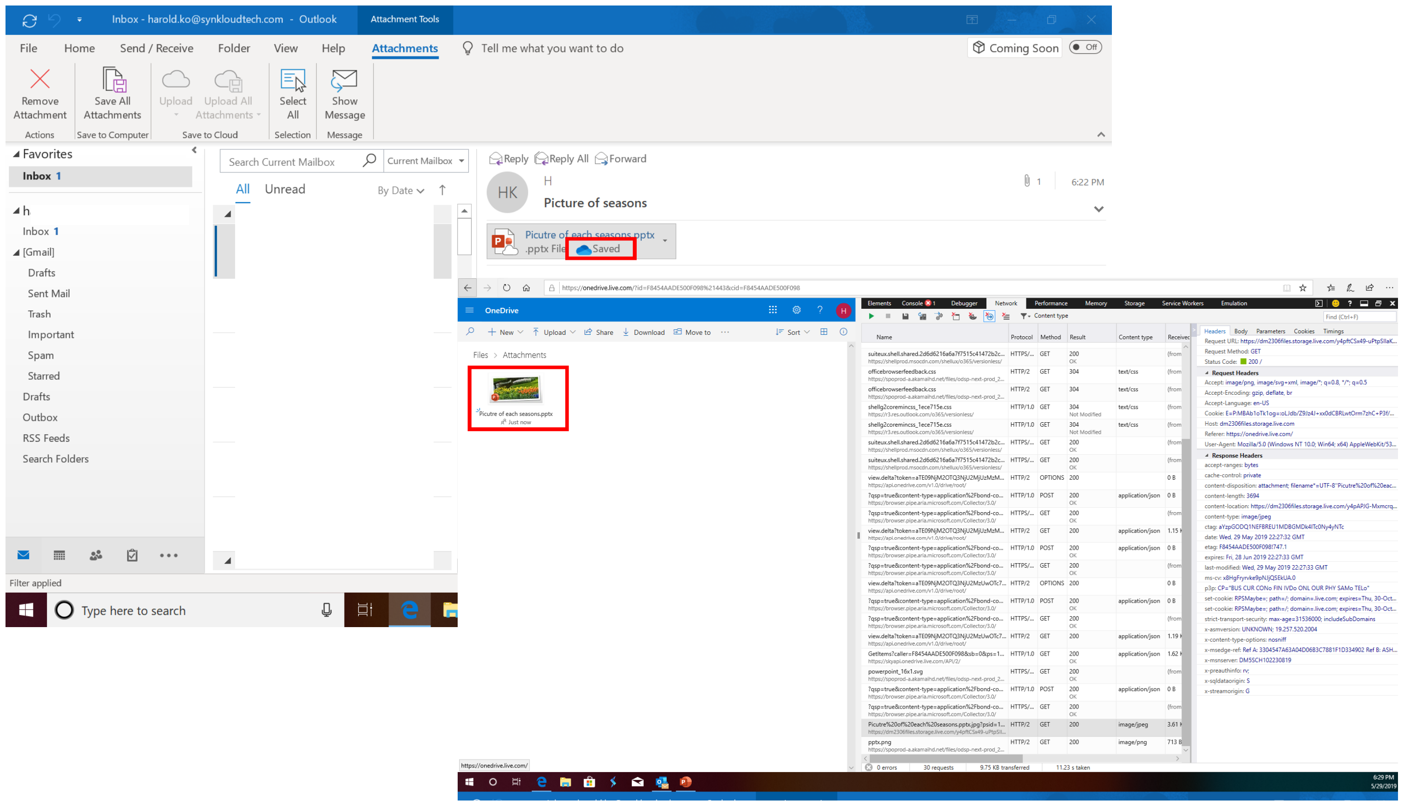
Claims of the '526 Patent	HP Device Access to Remote Storage Space
	[3rd step] completed "saving to OneDrive" 

EXHIBIT M

Exhibit 4

Exhibit 4 – Claim Chart for U.S. Patent No. 10,015,254

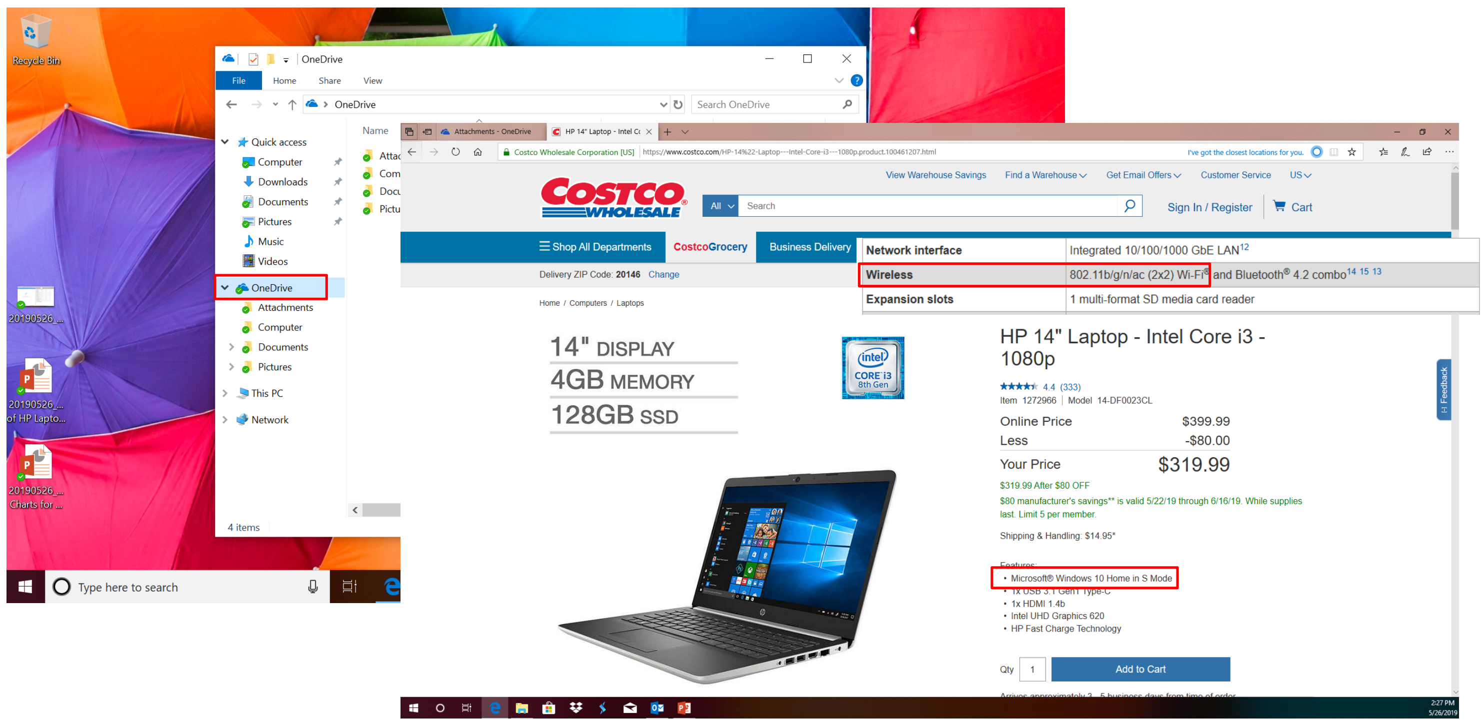
Claims of the '254 Patent	HP Device Access to Remote Storage Space
1. A wireless device accessing a remote storage space, the wireless device comprising:	A wireless device such as HP Laptop, e.g., HP 14" Laptop – Intel Core i3 with Windows OS and 802.11 b/g/n/ac WiFi, can access a remote storage space provided by storage service providers such as pre-installed Microsoft OneDrive ("OneDrive"), the HP Laptop comprising: 

Exhibit 4 – Claim Chart for U.S. Patent No. 10,015,254

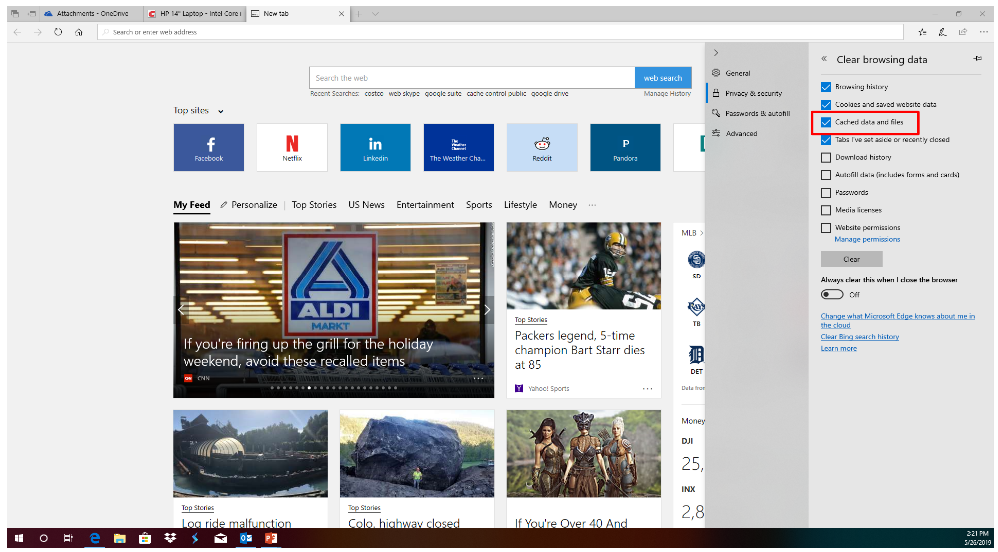
Claims of the '254 Patent	HP Device Access to Remote Storage Space
at least one cache storage for caching data received from the internet, and	HP Laptop comprises at least one cache storage, e.g., a cache storage configured for caching data and files such as for Microsoft Edge web browser: 
one computer-readable storage device comprising program instructions which, when executed by the wireless device, configure the wireless device accessing the remote space, wherein the program instructions comprise:	HP Laptop has at least one computer-readable storage device, e.g., Drive C:, comprising program instructions which, being executed by the HP Laptop, cause the HP Laptop to access to remote storage space, e.g., access to storage space provided by OneDrive, the program instructions comprise:

Exhibit 4 – Claim Chart for U.S. Patent No. 10,015,254

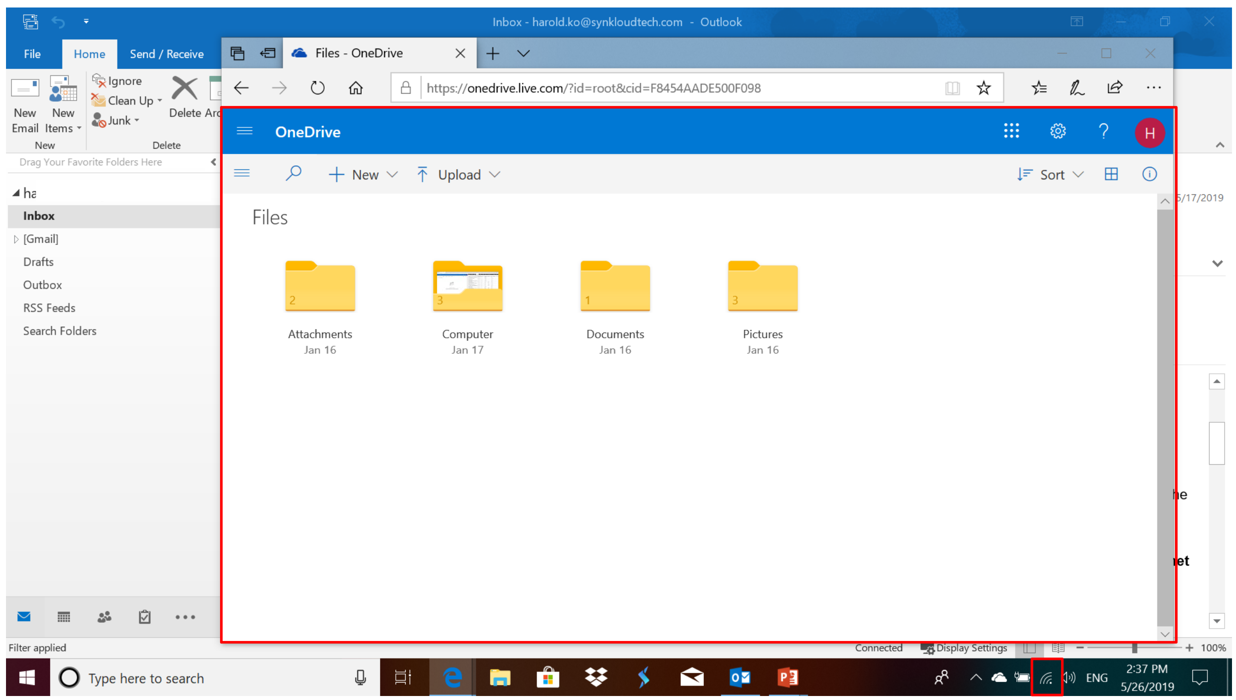
Claims of the '254 Patent	HP Device Access to Remote Storage Space
program instructions for wireless device establishing a communication link for accessing the remote storage space served by a first server;	Certain program executed on the HP Laptop establish a communication link, e.g., WiFi, for the wireless device to access to the storage space provided by a first server, e.g., OneDrive server, via Edge web browser that is illustrated in a screenshot taken from the HP Laptop. The OneDrive server allocates a 5 GB free storage space to a user. 

Exhibit 4 – Claim Chart for U.S. Patent No. 10,015,254

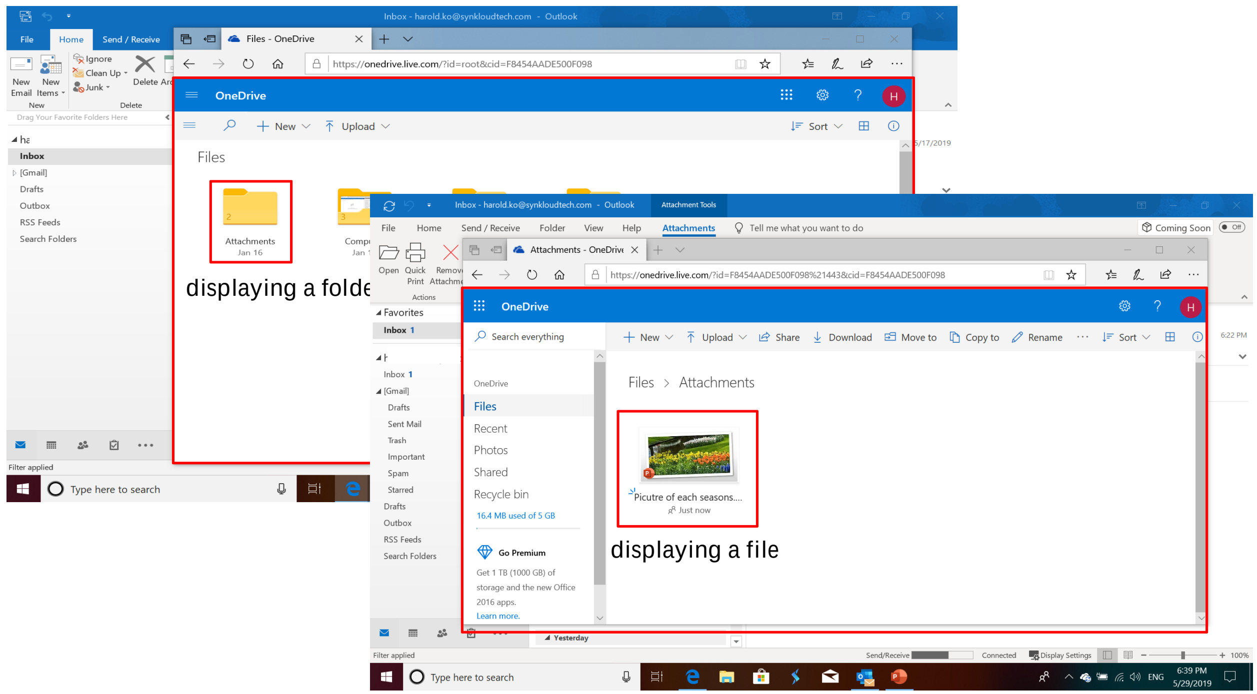
Claims of the '254 Patent	HP Device Access to Remote Storage Space
program instructions for the wireless device displaying the remote storage space upon receiving information of the remote storage space from the first server, and	Certain program instruction executed on the HP Laptop display the remote storage space, e.g., files or folders stored in the OneDrive, upon receiving information from the first server, e.g., OneDrive server. 

Exhibit 4 – Claim Chart for U.S. Patent No. 10,015,254

<table>
<tr><th>Claims of the '254 Patent</th><th>HP Device Access to Remote Storage Space</th></tr>
<tr><td>program instructions for the wireless device coupling with the first server to carry out a requested operation for accessing the remote storage space in response to a user, through the remote storage space displayed on the wireless device, performing the operation,</td><td rowspan="2">Certain program instructions executed couple the HP Laptop with the storage server of the OneDrive across the wireless link to carry out a requested operation for remotely accessing to the assigned storage space.

The requested operation for the access to the storage space comprises storing data into the remote storage space or retrieving data from the remote storage space, e.g., uploading a file into the remote storage space (storing), or downloading a file from the remote storage space into the HP Laptop (retrieving):

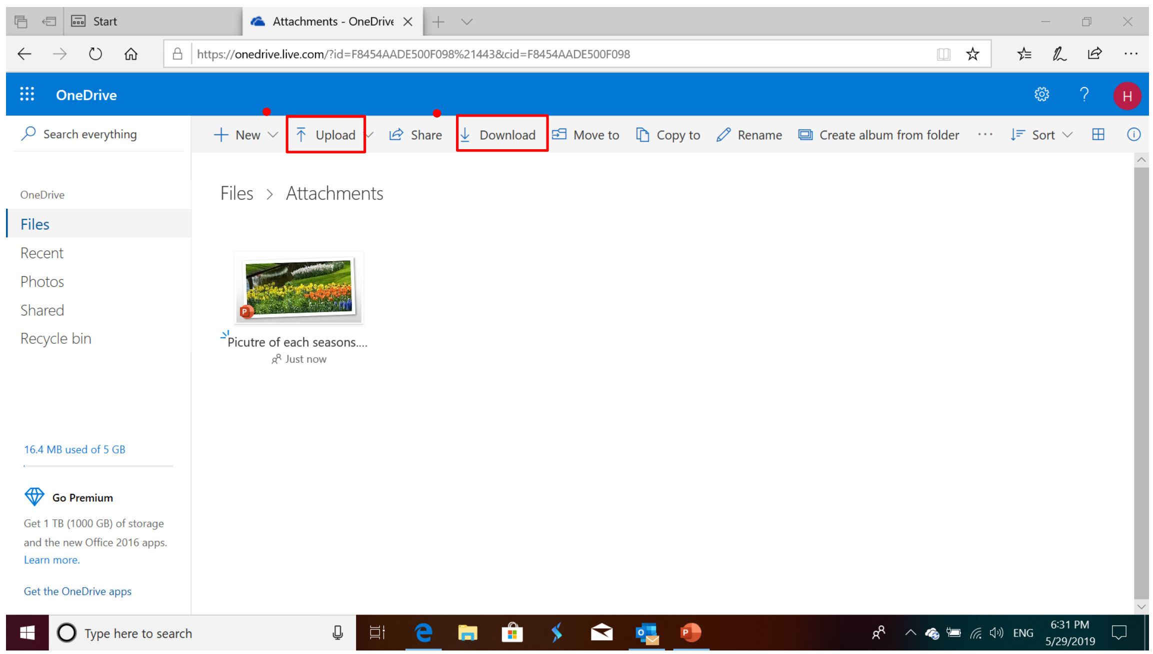

</td></tr>
<tr><td>wherein the operation being carried out for accessing the remote storage space comprises from the wireless device storing data therein or retrieving data therefrom,</td></tr>
</table>

Exhibit 4 – Claim Chart for U.S. Patent No. 10,015,254

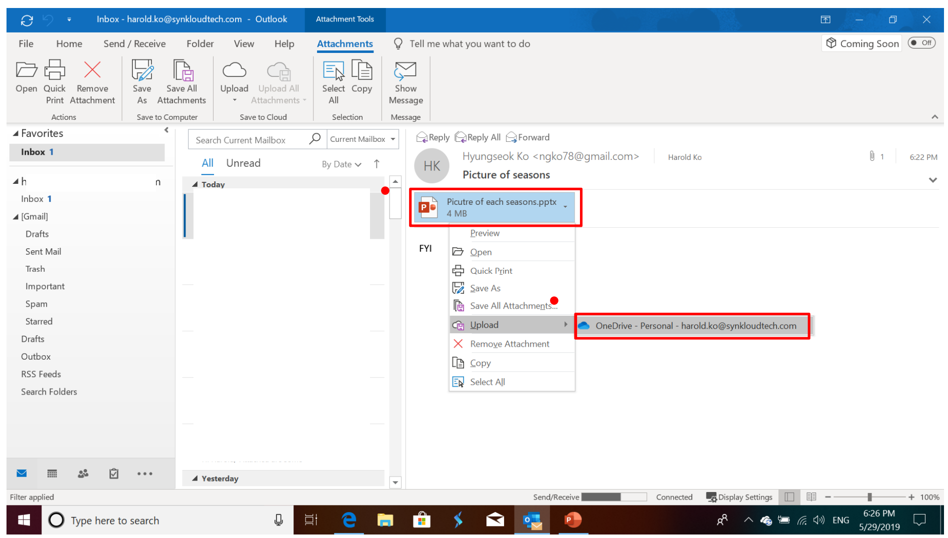
Claims of the '254 Patent	HP Device Access to Remote Storage Space
the storing data comprising to download a file from a second server across a network into the storage space through utilizing information for the file cached in the cached storage in the wireless device.	The storing data comprises to download a file from a second server, e.g., from a remote Outlook Server, into the user's assigned storage space of the OneDrive by using download information for the file cached in the cache storage of the HP Laptop. The next screenshot illustrates such downloading a file, e.g. downloading a 2MB PowerPoint file, from an Outlook Server into the user's assigned storage space of OneDrive. • Downloading a PowerPoint file from an Outlook Server to OneDrive [1st step] choose a file to upload and select "Upload to OneDrive" 

Exhibit 4 – Claim Chart for U.S. Patent No. 10,015,254

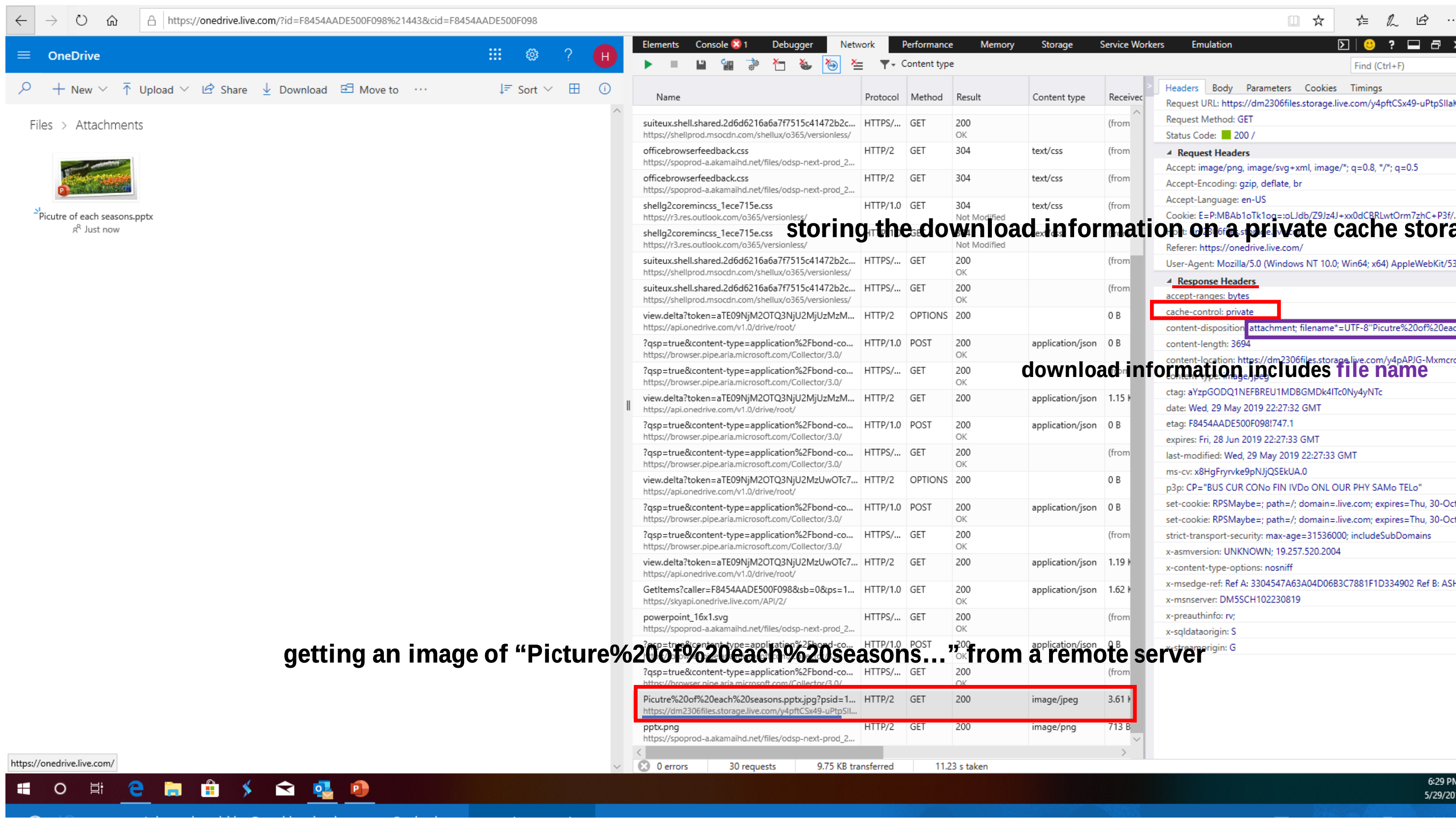
Claims of the '254 Patent	HP Device Access to Remote Storage Space
	[2nd step] utilizing information for the file cached in the cache storage 

Exhibit 4 – Claim Chart for U.S. Patent No. 10,015,254

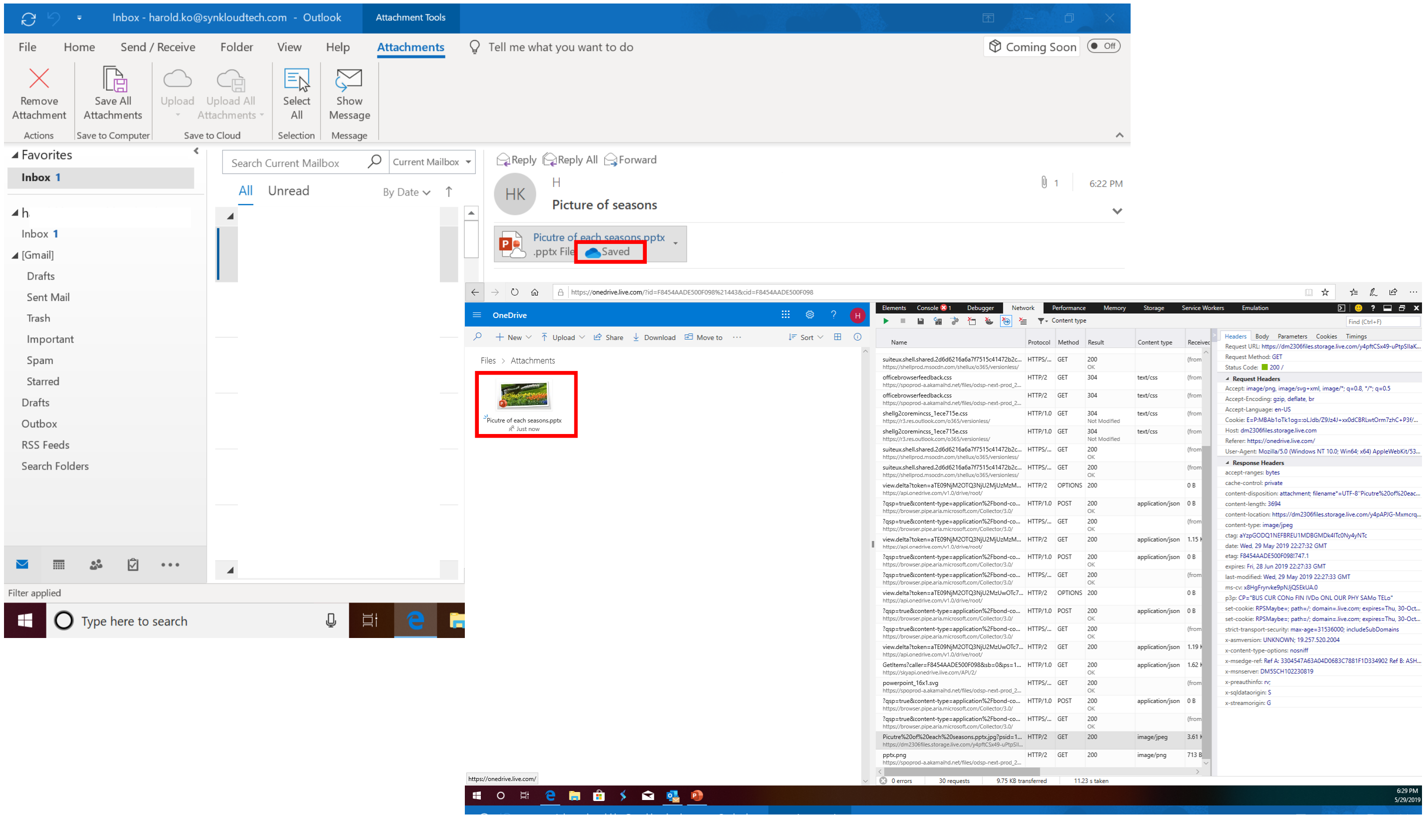
Claims of the '254 Patent	HP Device Access to Remote Storage Space
	[3rd step] completed "saving to OneDrive" 

EXHIBIT N

Exhibit 7

Exhibit 7 – Claim Chart for U.S. Patent No. 7,870,225

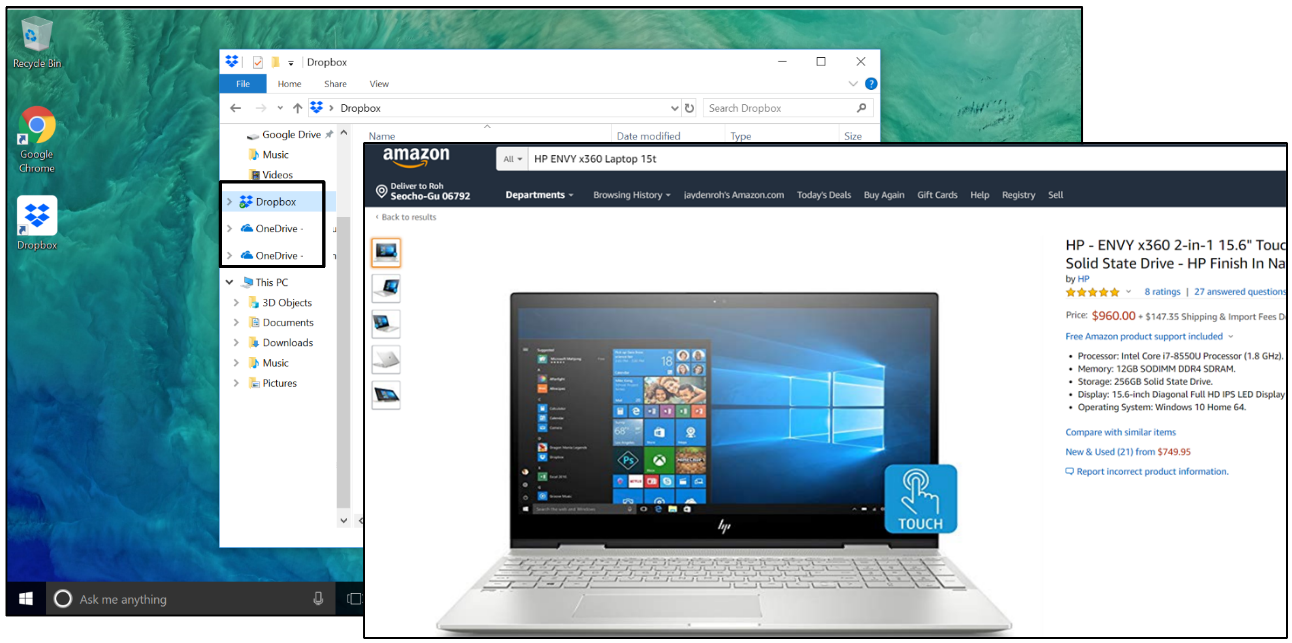
Claims of the '225 Patent	HP Device Access to Remote Storage Space
1. A network-attached device (NAD) access system wherein a host, having an internal host systembus and running an operating system, controls an external device through a carrying general-purpose network traffic using a certain network protocol, the system comprising:	A host (e.g., HP Envy x360 Laptop) having an internal host bus and running an operating system (e.g., MS Windows 10 Home (64-Bit)) is configured to control an external device through carrying general-purpose network traffic using a certain network protocol (e.g., 802.11.b,g) the system comprising: 

Exhibit 7 – Claim Chart for U.S. Patent No. 7,870,225

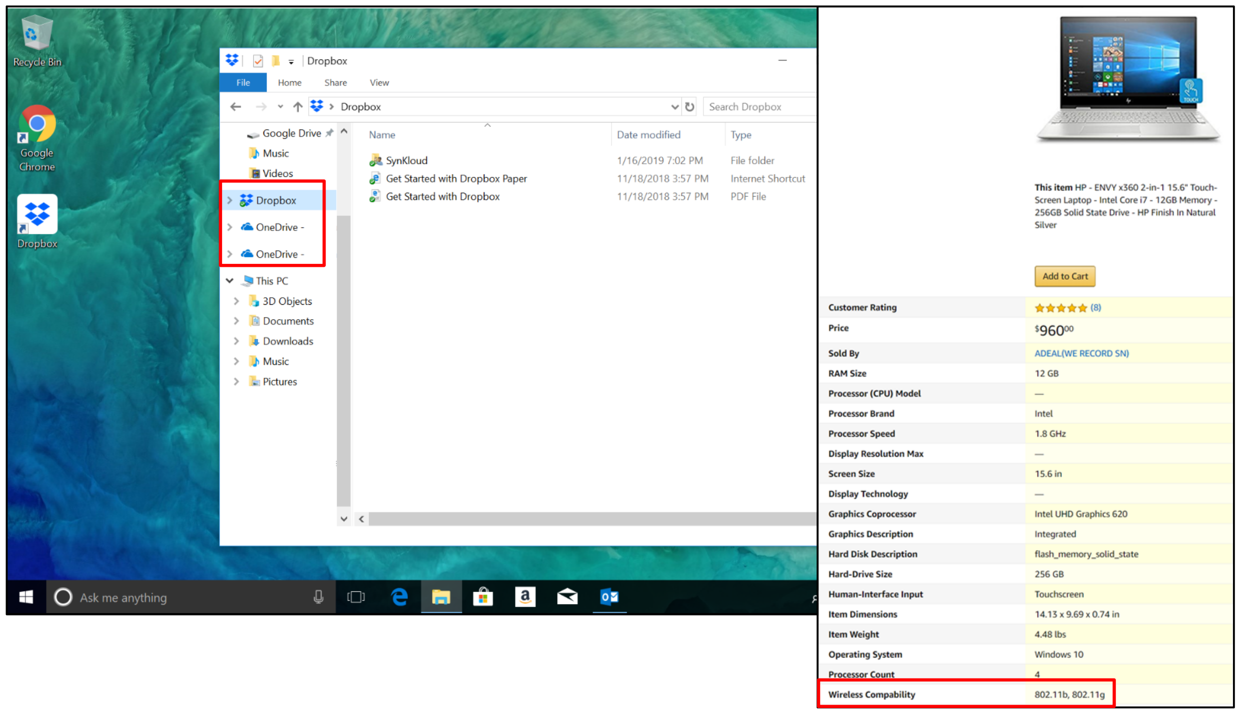
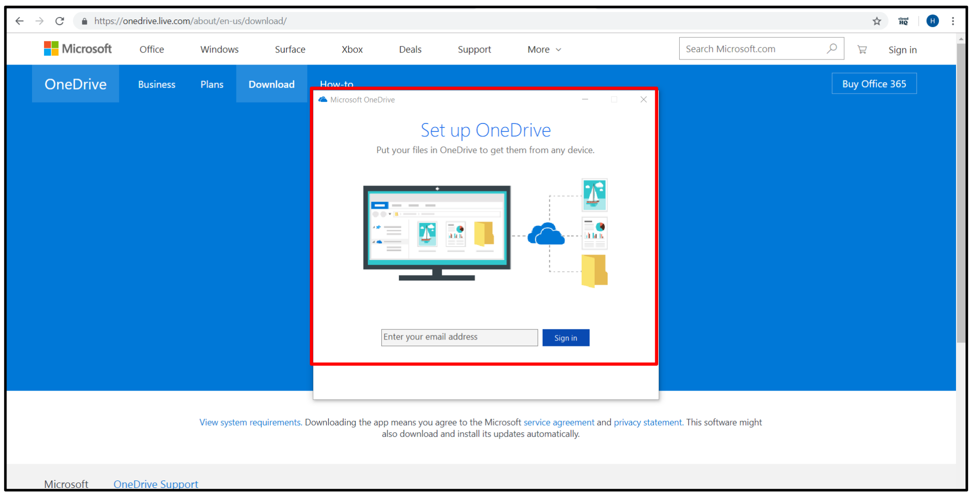
Claims of the '225 Patent	HP Device Access to Remote Storage Space
a network interface card (NIC) installed at the host for providing a general purpose network connection between the host and the network and via the network to other devices coupled the network;	The host has a network interface card (NIC) (e.g., a WiFi Adapter for 802.11.b,g) for providing a general purpose network connection between the host (the HP Laptop) and the network and via the network to other devices coupled to the network. 

Exhibit 7 – Claim Chart for U.S. Patent No. 7,870,225

Claims of the '225 Patent	HP Device Access to Remote Storage Space
a network-attached device (NAD) having a data storage to store data, the NAD coupled to the network for receiving device level access commands from the host in data link frames according to the certain network protocol through the network; and	The network-attached device (NAD) having a storage (e.g., a free 5GB of storage space allocated to each of the users by the server of MS OneDrive) to store data (e.g., documents, songs, pictures, or moving pictures, etc.) couples to the network for receiving device level access commands (e.g., associated with request for signing into OneDrive or request for uploading files) from the host in data link frames. • device level access commands : associated and used with the request for signing into OneDrive 

Case 1:19-cv-01360-RGA Document 15-1 Filed 11/12/19 Page 93 of 120 PageID #: 221

Exhibit 7 – Claim Chart for U.S. Patent No. 7,870,225

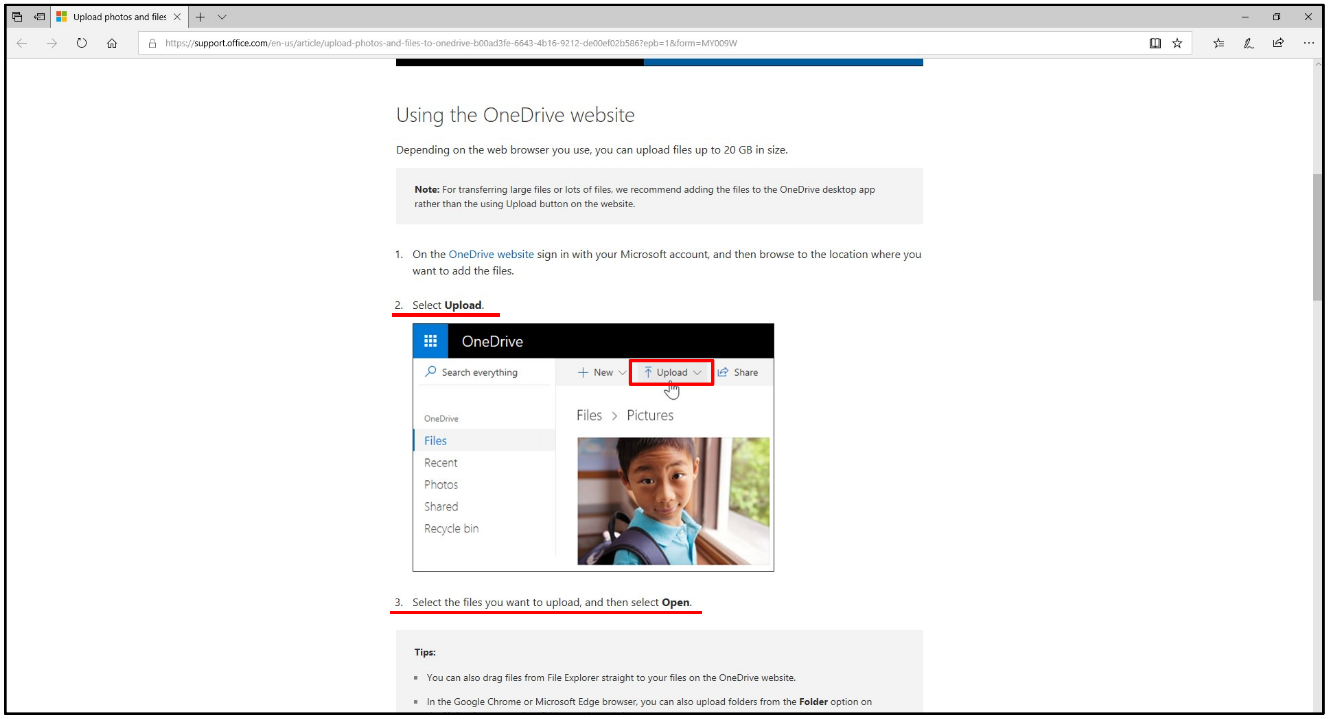
Claims of the '225 Patent	HP Device Access to Remote Storage Space
	• device level access commands : associated and used with the request for uploading files into OneDrive 

Exhibit 7 – Claim Chart for U.S. Patent No. 7,870,225

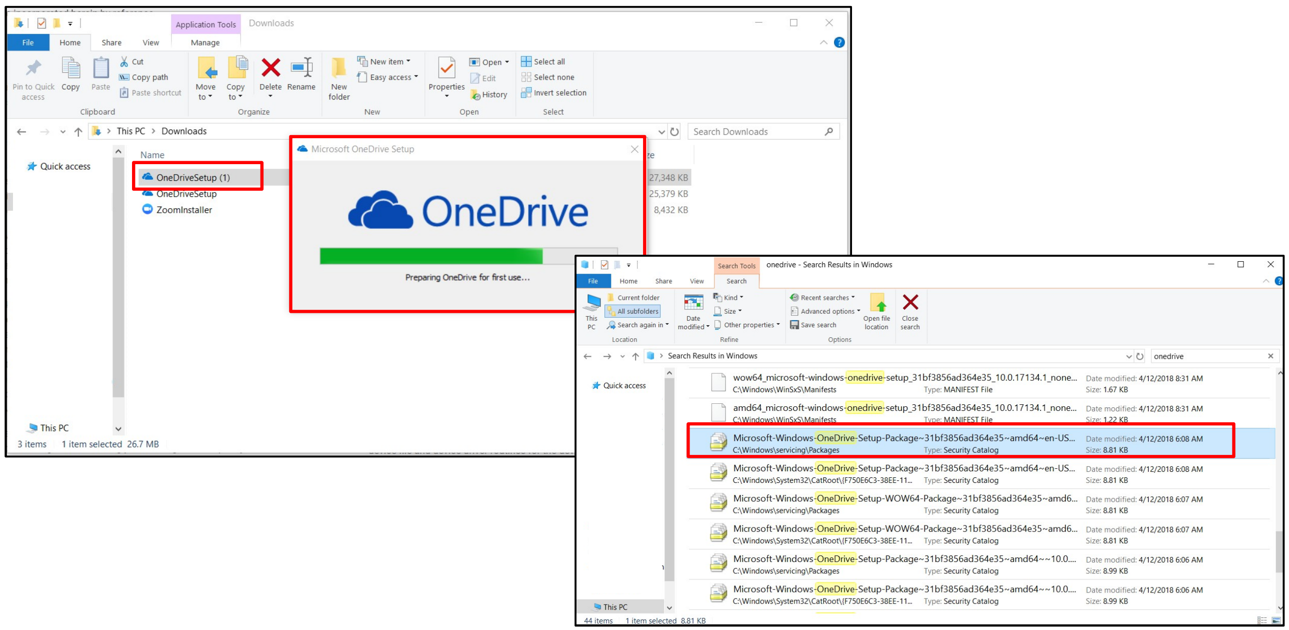
Claims of the '225 Patent	HP Device Access to Remote Storage Space
a device driver, running at the host, for creating a virtual host bus adapter in software controlling the NAD through the network via the NIC, the device driver enumerating NAD that are available over the network, not directly attached to thehost internal system bus, to make the host recognize the NAD as a host local device;	The installed or activated device driver (e.g., OneDriveSetup.exe, Microsoft-Windows-OneDrive-Setup-Package) running on the host (i.e., HP Laptop) enumerates the NAD (e.g., the remote storage space of OneDrive) which is available over the network via the NIC (e.g., a WiFi Adapter for 802.11.b,g). • installing or activating a device driver of OneDrive 

Exhibit 7 – Claim Chart for U.S. Patent No. 7,870,225

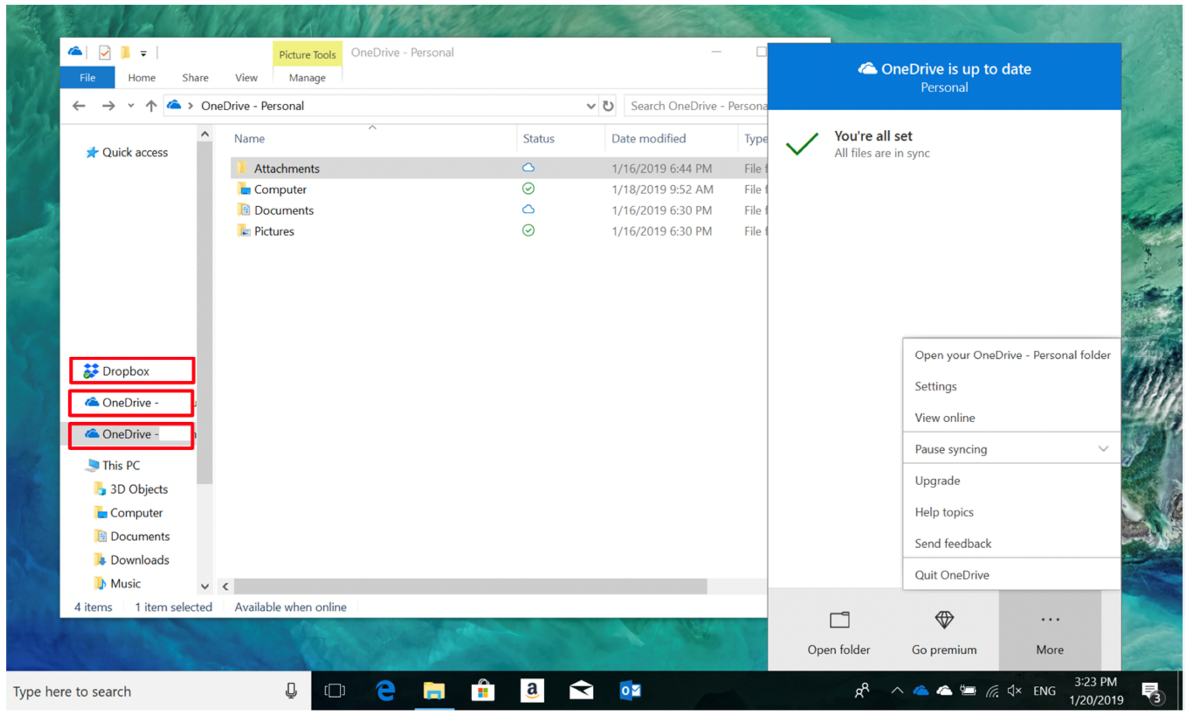
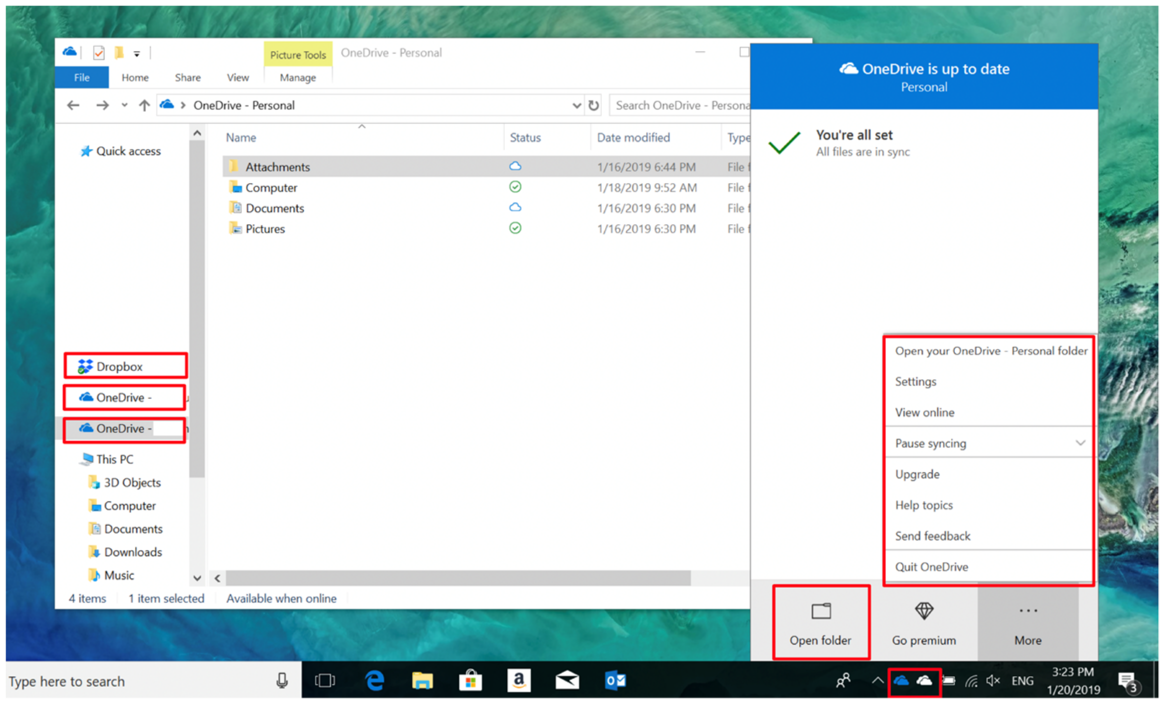
Claims of the '225 Patent	HP Device Access to Remote Storage Space
	• NAD, the remote storage server of OneDrive, that are available is enumerated  Although the NAD, the remote storage space of OneDrive, is not directly attached to the host internal system, it is recognized as host local device because • it is displayed as if it is one of the local devices, and • user can access the file within the remote storage.

Exhibit 7 – Claim Chart for U.S. Patent No. 7,870,225

Claims of the '225 Patent	HP Device Access to Remote Storage Space
	The device driver creates a virtual host bus adapter which controls the NAD (e.g., the remote storage space of the OneDrive) through the network (e.g., "Open folder", " Open your OneDrive – Personal folder", "Settings", "View online", "Pause syncing", etc.). 

Case 1:19-cv-01960-RGA Document 15-1 Filed 11/12/19 Page 97 of 120 PageID #: 225

Exhibit 7 – Claim Chart for U.S. Patent No. 7,870,225

Claims of the '225 Patent	HP Device Access to Remote Storage Space
the virtual host bus adapter controlling the NAD in a way indistinguishable from the way as a physical host bus adapter device controls device so that the host recognizes the NAD as if it is a local device connected directly to the system bus of the host.	The virtual host bus adapter controlling the NAD (e.g., the remote storage space of OneDrive) in a way indistinguishable from the way as a physical host bus adapter device controls device, i.e., the user can access to the remote storage space of OneDrive and open the folder and file within the storage space through file explorer, so that the host recognizes the OneDrive space as if it is a local device.